

Cranfield University

Gokop Goteng

Development of a Grid Service for Multi-objective Design
Optimisation

School of Applied Sciences

PhD

Cranfield University

Academic Year 2008-2009

PhD THESIS

April 2009

Gokop Goteng

Development of a Grid Service for Multi-objective Design
Optimisation

Supervisors: Dr. Ashutosh Tiwari and Prof. Rajkumar Roy

Academic Year 2008 to 2009

This thesis is submitted in partial fulfilment of the requirements
for the degree of doctor of philosophy

© Cranfield University, 2009. All rights reserved. No part of this publication may be
reproduced without the written permission of the copyright holder.

Abstract

The emerging grid technology is receiving great attention from researchers and applications that need computational and data capabilities to enhance performance and efficiency. Multi-Objective Design Optimisation (MODO) is computationally and data challenging. The challenges become even more with the emergence of evolutionary computing (EC) techniques which produce multiple solutions in a single simulation run. Other challenges are the complexity in mathematical models and multidisciplinary involvement of experts, thus making MODO collaborative and interactive in nature. These challenges call for a problem solving environment (PSE) that can provide computational and optimisation resources to MODO experts as services. Current PSEs provide only the technical specifications of the services which is used by programmers and do not have service specifications for designers that use the system to support design optimisation as services. There is need for PSEs to have service specification document that describes how the services are provided to the end users. Additionally, providing MODO resources as services enabled designers to share resources that they do not have through service subscription.

The aim of this research is to develop specifications and architecture of a grid service for MODO. The specifications provide the service use cases that are used to build MODO services. A service specification document is proposed and this enables service providers to follow a process for providing services to end users.

In this research, literature was reviewed and industry survey conducted. This was followed by the design, development, case study and validation. The research studied related PSEs in literature and industry to come up with a service specification document that captures the process for grid service definition. This specification was used to develop a framework for MODO applications. An architecture based on this framework was proposed and implemented as DECGrid (Decision Engineering Centre Grid) prototype.

Three real-life case studies were used to validate the prototype. The results obtained compared favourably with the results in literature. Different scenarios for using the services among distributed design experts demonstrated the computational synergy and efficiency in collaboration. The mathematical model building service and optimisation service enabled designers to collaboratively build models using the collaboration service. This helps designers without optimisation knowledge to perform optimisation.

The key contributions in this research are the service specifications that support MODO, the framework developed which provides the process for defining the services and the architecture used to implement the framework. The key limitations of the research are the use of only engineering design optimisation case studies and the prototype is not tested in industry.

Dedication

The fear of the **Lord** is the beginning of knowledge [*Proverbs 1:7*]. I wish to dedicate this research to **Almighty God** who has sustained me through out and given me the knowledge to carry out the research.

Acknowledgements

I wish to thank my two supervisors Dr. Ashutosh Tiwari and Prof. Rajkumar Roy for their support and guidance throughout the research period. Their invaluable encouragement and inputs to ensure that the research is completed within schedule has been rewarding.

I am highly indebted to my mother, Nakup, who toiled to see me become what I am with the help of her late brother, Ezekiel Gokir, whom I owed a lot and wish he were alive to see my graduation.

Worthy of a million thanks are my sponsors, the Petroleum Technology Development Fund (PTDF) Nigeria and Cranfield University. Without their support, this research would not have been possible.

I also wish to thank Prof. John Kay for his assistance each time I run into some difficulties as an international student. He is ever ready to help with resources that I need for research.

I wish to acknowledge my subject assessor, Prof. Mark Savill, who has been encouraging all the time. His objective observations on how to improve the research has been rewarding.

I thank Emanuela and Teresa for their usual help when I needed any secretariat work done. They are wonderful secretaries.

I wish to thank Ian Osborn, the Managing Director of Intellect-Grid Computing Now! and his ICT Manager, Tara Kelly, who saw to my internship at National e-Science Centre (NeSC) in University of Edinburgh, Scotland. The internship at NeSC forms part of my industry survey.

I wish to thank Jerry Fishenden, the Microsoft Technology Officer in UK who provided the opportunity for me to attend the Microsoft European Innovation Day in Brussels, Belgium. The occasion afforded me the opportunity to present my winning concept on using grid computing to counter terrorism to attendees.

Also worthy of mention is Dr. Dave Berry who took me round and introduced me to members of NeSC during my internship. He ensured I got all the necessary support and materials for my internship.

I wish to acknowledge the valuable inputs made by Dhish Saxena. Dhish put me through the working principles of NSGA-II algorithms.

I wish to acknowledge the interest and help Elhadj Benkhelifa offered to me which helped me to improve on the validation process of the research. Finally I wish to thank all researchers and PhD students in Manufacturing Department for providing useful suggestions on how to improve the research.

List of Publications

Goteng, G., Tiwari, A. and Roy, R. (2007). Grid Computing for Engineering Optimisation and Simulation: Applications for Oil & Gas Reservoir Exploration. *Petroleum Training Journal (PTJ)*, vol. 4, no. 1, p. 52-66, ISSN: 1595-9104.

Goteng, G., Tiwari, A. and Roy, R. (2008). Virtualisation of Grid Services for Multidisciplinary Optimisation in Aerospace Industry. In: *Cranfield Multi-Strand Conference: Creating Wealth Through Research and Innovation (CMC-2008)*, Cranfield University, Bedfordshire, UK, 6-7 May, 6 pages.

Goteng, G., Tiwari, A. and Roy, R. (2009). Grid Services for Multi-objective Optimisation. In: *CIRP Design Conference: (CIRP-2009)*, Cranfield University, Bedfordshire, UK, March 30-31, p. 73-77.

Goteng, G., Tiwari, A. and Roy, R. (2009). Grid Computing: Combating Global Terrorism with the World Wide Grid. *Handbook of Research on Grid Technologies and Utility Computing Concepts for Managing Large-scale Applications*, ISBN: 978-1-60566-184-1, p. 265-280.

Roy, R., Tiwari, A., Tafesse-Azene, Y. and Goteng, G. (2008). Evolutionary Computing in Engineering Design. In: Laha, D. and Mandal, P. (eds.). *Handbook of computational intelligence in manufacturing and production management*, ISBN: 978-1-59904-582-5, 167-184, Information Science Reference, USA.

Tiwari, A., Goteng, G. and Roy, R. (2007). Evolutionary Computing Within Grid Environment. *Advances in Evolutionary Computing for System Design*: Vol. 66, p. 229-248.

Submitted Papers

Goteng, G., Tiwari, A. and Roy, R. (2008). Grid Computing for Multidisciplinary Optimisation: Evolution and Future Trends. Submitted to *ACM Computing Surveys Journal*.

Goteng, G., Tiwari, A. and Roy, R. (2008). Grid Computing: A Comparison of Industry Applications with Research Efforts. Submitted to *International Journal of High Performance Computing and Applications (IJHPCA)*.

Goteng, G., Tiwari, A. and Roy, R. (2008). Grid Computing-Enabled Engineering Design Optimisation: Problem Solving Environments for Concurrent Engineering. Submitted to *Future Generations Computing Systems Journal*, Elsevier Publications.

Award

Goteng, G. (2006). **Grid Computing: Combating Global Terrorism with the World Wide Grid**. A Nation Wide (UK) Grid Computing competition organised by *DTI-Intellect and British Computer Society (BCS)* and sponsored by *Microsoft and Intel* on the 28th September 2006. This project won the first prize.

Table of Contents

Abstract.....	i
Dedication.....	ii
Acknowledgements	iii
List of Publications.....	iv
Table of Contents	v
List of Figures.....	x
List of Tables.....	xiii
Normenclature	xiv
Chapter 1 - Introduction	1
1.1 Overview of the research.....	1
1.2 Introduction to grid services.....	3
1.3 Introduction to multi-objective design optimisation (MODO).....	5
1.4 Motivation for grid services in MODO applications.....	6
1.4.1 Computational motivation	6
1.4.2 Data motivation	7
1.4.3 Multidisciplinary and collaboration motivation	8
1.5 Key variables that affect collaboration.....	8
1.5.1 Security.....	9
1.5.2 Autonomy	9
1.5.3 Performance.....	10
1.5.4 Hosting environment	10
1.5.5 Quality of service.....	10
1.5.6 Reliability	11
1.5.7 Scalability and interoperability.....	11
1.5.8 Transparency	11
1.6 Scope of the research.....	11
1.7 Structure of the Thesis.....	12
1.8 Summary.....	13
Chapter 2 - Review of Literature.....	14
2.1 Introduction	14
2.2 Grid computing.....	15
2.3 Generations of grid computing.....	16
2.3.1 First generation of grid computing	17
2.3.2 Second generation of grid computing.....	20
2.3.3 Third generation of grid computing	22
2.3.4 Fourth generation of grid computing.....	26
2.3.5 Future generations of grid computing	27
2.4 Grid projects	31
2.5 Multi-objective design optimisation.....	32
2.5.1 Engineering design optimisation philosophy	32
2.5.2 Challenges in MODO	33
2.5.3 MODO methods and approaches.....	34

2.5.4	Adoption of evolutionary computing approach.....	37
2.5.5	MODO service approach.....	40
2.5.6	Grid as an enabler for MODO.....	40
2.6	Grid-enabled problem solving environments.....	42
2.6.1	Geodise.....	42
2.6.2	FIPER.....	43
2.6.3	SORCER.....	44
2.6.4	DAME.....	44
2.6.5	Globus toolkit.....	45
2.6.6	Challenges that PSEs face in solving MODO problems.....	46
2.7	Grid service definition.....	47
2.7.1	What is a grid service.....	48
2.7.2	How to define a grid service.....	49
2.8	Critical variables that make grid service practically useful.....	56
2.8.1	Economic variables.....	56
2.8.2	Technological variables.....	58
2.8.3	Operational variables.....	59
2.9	Facilities to develop MODO services.....	59
2.9.1	Middleware.....	60
2.9.2	Portals.....	60
2.9.3	Certificate authority.....	60
2.9.4	Schedulers.....	61
2.9.5	Resource brokers.....	61
2.9.6	Operating systems.....	62
2.10	Gaps in literature.....	62
2.11	Summary.....	63
Chapter 3 - Research Aim, Objectives and Methodology.....		64
3.1	Aim of the research.....	64
3.2	Objectives of the research.....	64
3.3	Research methodology.....	65
3.3.1	Literature review.....	65
3.3.2	Industry survey.....	65
3.3.3	Grid services specifications and design.....	66
3.3.4	Grid architecture for MODO service.....	67
3.3.5	Implementation.....	67
3.3.6	Case study and validation.....	67
3.4	Reasons for adopting the above methodology.....	70
3.5	Summary.....	72
Chapter 4 - Grid in Industry.....		73
4.1	Industry survey.....	73
4.1.1	Aim and objectives of the survey.....	73
4.1.2	Industry survey methodology.....	74
4.2	DTI-Intellect: Grid Computing Now! Industry internship.....	81
4.3	Industry survey findings.....	83
4.3.1	Grid applications issues.....	83
4.3.2	Grid implementation issues.....	87
4.4	Gaps between academia and industry.....	92
4.5	How literature and industry survey guided the design and implementation ..	94

4.6	Summary.....	95
Chapter 5 - Grid Service Specifications and Design for MODO		96
5.1	MODO service design concept.....	96
5.1.1	Specifications of MODO service.....	98
5.1.2	Different models to deliver MODO services.....	98
5.1.3	Dimensions of a service.....	99
5.1.4	Quality of MODO service	100
5.2	Analysis of service specifications.....	101
5.2.1	Globus service specifications	102
5.2.2	FIPER service specifications	103
5.2.3	SORCER service specifications	104
5.2.4	Geodise service specifications.....	105
5.3	MODO services design.....	106
5.3.1	Design of mathematical model building service	106
5.3.2	Design of qualitative mathematical model building service	110
5.3.3	Design of collaboration service specification.....	113
5.3.4	Design of compute service specification	115
5.3.5	Optimisation parameter input service.....	117
5.3.6	Resource aggregation	118
5.4	MODO interface design.....	120
5.4.1	Definition of MODO service interface for domain disciplines	121
5.4.2	Definition of MODO service interface for users	121
5.5	Research contribution in service specification	127
5.5.1	MODO service level agreement	130
5.5.2	MODO resource registry	131
5.5.3	MODO aggregator source	131
5.5.4	MODO search strategy interface	132
5.5.5	MODO functional requirements.....	132
5.5.6	MODO services	132
5.6	Summary.....	132
Chapter 6 - Service Framework and Architecture of DECGrid		134
6.1	Transition from design to actual framework	134
6.2	Process of grid service definition	135
6.2.1	Identification of service requestors	136
6.2.2	Choice of grid type to publish services	136
6.2.3	Definition of grid service requirements.....	136
6.2.4	Identification of existing competing grid services	137
6.2.5	Grid convention platforms development	137
6.2.6	Policy and agreement between grid service providers and requestors .	138
6.2.7	Grid Service definition implementation	138
6.3	Service framework.....	139
6.3.1	MODO grid service framework.....	140
6.4	DECGrid architecture	143
6.5	Research contributions of the framework and architecture	145
6.6	Summary.....	145
Chapter 7 - Implementation of DECGrid and MODO Services.....		146
7.1	Preparatory statement	146
7.2	Facility and tools	147

7.3	Setting-up the grid infrastructure.....	148
7.4	Case study considerations.....	150
7.5	Services implementation.....	151
7.5.1	How to write a service	154
7.5.2	Mathematical model building service.....	156
7.5.3	Search strategy selection	160
7.5.4	Collaboration service.....	164
7.5.5	NSGA-II input parameter and optimisation service.....	167
7.5.6	Qualitative optimisation and visualisation services.....	171
7.5.7	Computational service	172
7.6	Implementation issues	175
7.7	Summary.....	177
Chapter 8 - Validation using Case Studies		178
8.1	Real-life case studies	178
8.1.1	Aerospace: Design of a turbine blade cooling system.....	178
8.1.2	Manufacturing: Welded beam problem.....	180
8.1.3	Architecture: Design of a manufacturing plant layout and floor planning 181	
8.2	Reasons for choosing the case studies	183
8.3	Validation methodology	184
8.3.1	What to validate.....	184
8.3.2	How to validate.....	185
8.3.3	What is the use case scenario	185
8.3.4	Who participates in validation.....	186
8.4	Design of gas turbine blade cooling system	187
8.4.1	Experimental results	192
8.5	Design of welded beam problem.....	200
8.5.1	Experimental results	202
8.6	Design of a manufacturing plant layout and floor planning.....	204
8.6.1	Experimental results	208
8.7	Validation of results	214
8.7.1	Why grid?	215
8.7.2	Design of turbine blade cooling system	218
8.7.3	Design of the welded beam problem.....	219
8.7.4	Design of a manufacturing plant layout and floor planning.....	220
8.7.5	Validation questionnaire discussions	221
8.8	Summary.....	221
Chapter 9 - Discussion and Conclusions		222
9.1	Discussion.....	222
9.1.1	Literature review.....	223
9.1.2	Industrial context	224
9.1.3	Gaps in literature and industry.....	226
9.1.4	Design of DECGrid	227
9.1.5	Implementation of DECGrid	229
9.1.6	Validation using real-life case studies	232
9.1.7	What benefits will business get from the system	233
9.1.8	Pending issues	233
9.1.9	Main contributions.....	234

9.1.10	Limitations of the research	236
9.1.11	Future research issues	236
9.2	Conclusions	237
References	240
Appendix-I	255
Appendix-II	263
Appendix-III	273
Appendix-IV	277

List of Figures

Figure 1.1: Description of interactions in Service-Oriented Architecture.....	5
Figure 1.2: Process of mathematical model building, collaboration and optimisation services	8
Figure 2.1: Grid application areas and number of research papers	19
Figure 2.2: Grid Application Areas and Number of Research Papers.....	20
Figure 2.3: Middleware and percentage of papers	21
Figure 2.4: Service-oriented Models and Number of Papers	23
Figure 2.5: Web and Grid Convergence Process as the Generations of Grid Mature (Source: Foster <i>et al.</i> , 2001)	24
Figure 2.6: PSE pie chart showing PSE tools and Number of Papers/percentage	25
Figure 2.7: GA flowchart (Deb, 2001)	38
Figure 2.8: Simple Geodise Architecture (Cox <i>et al.</i> , 2001).....	43
Figure 2.9: Globus Layered Architecture (Foster and Kesselman, 1999).....	46
Figure 2.10: OGSA Grid Service (Foster <i>et al.</i> , 2002a)	54
Figure 3.1: Summary of methodology.....	66
Figure 3.2: Important sequence of research achievements and chapters.....	71
Figure 4.1: Introductory e-mail message for the online questionnaire.....	78
Figure 4.2: Grid computing incentives	84
Figure 4.3: Grid application features and components.....	85
Figure 4.4: Grid application disciplines	86
Figure 4.5: Grid middleware	87
Figure 4.6: Grid problem solving environments (FIPER=Federated Intelligent Problem Environment; SORCER=Service-Oriented Computing Environment; DAME=Distributed Aircraft Maintenance Environment).....	88
Figure 4.7: Operating systems for high performance computing.....	89
Figure 4.8: Limitations and challenges of grid implementation.....	91
Figure 4.9: Challenges of grid implementation	92
Figure 5.1: Globus service specifications.....	102
Figure 5.2: SORCER service spécification document (Sobolewski, 2007)	104
Figure 5.3: Geodise service specifications (Cox <i>et al.</i> , 2001)	105
Figure 5.4: Mathematical model building use case diagram	107
Figure 5.5: Specifications of quantitative mathematical model building service.....	108
Figure 5.6: Mathematical model building sequence diagram (Part 1)	109
Figure 5.7: Mathematical model building sequence diagram (Part 2)	110
Figure 5.8: Qualitative model use case diagram.....	111
Figure 5.9: Qualitative service specification sequence diagram	112
Figure 5.10: Collaboration service specification use case diagram.....	114
Figure 5.11: Collaborative service specification sequence diagram	114
Figure 5.12: Compute service specification use case diagram	115
Figure 5.13: Compute service specification sequence diagram	116
Figure 5.14: Optimisation parameter input use case diagram	117
Figure 5.15: Optimisation parameter input sequence diagram.....	118
Figure 5.16: MODO resource aggregation sequence diagram	119
Figure 5.17: Explicit model generation	122

Figure 5.18: NSGA-II input parameters (R_Var=Real Variable) (phase 1)	122
Figure 5.19: Boundary values for real variables.....	124
Figure 5.20: NSGA-II input parameters (phase 2)	124
Figure 5.21: Properties of binary variables	125
Figure 5.22: NSGA-II input parameters (phase 3)	125
Figure 5.23: Implicit model generation	126
Figure 5.24: Tree diagram of the services and interfaces.....	127
Figure 5.25: MODO service specification.....	129
Figure 6.1: Process of grid service definition.....	135
Figure 6.2: DECGrid MODO service framework	141
Figure 6.3: DECGrid architecture	144
Figure 7.1: Main domain selection interface for distributed users	151
Figure 7.2: Factory and instance patterns of MODO resource creation.....	154
Figure 7.3: Mathematical model interface definition codes	155
Figure 7.4: Optimisation data schema	157
Figure 7.5: Explicit model building interface.....	158
Figure 7.6: Implicit math model interface	160
Figure 7.7: First stage of search strategy selection optimisation resource query process (end point =EP; resource property=RP; uniform resource locator=URL)	162
Figure 7.8: Search strategy selection process.....	163
Figure 7.9: Seven servers display and prompt for password to have access to a host .	165
Figure 7.10: MODO tables in MODODatabase	166
Figure 7.11: Virtual sharing of computational resources	167
Figure 7.12: First input parameter service interface.....	168
Figure 7.13: Second input parameter interface.....	169
Figure 7.14: Third input parameter interface.....	170
Figure 7.15: Description of math model building and parameter inputs for optimisation	171
Figure 7.16: Condor job submission description file	173
Figure 7.17: Job submission definition file	175
Figure 8.1: General arrangement of five-pass cooling of turbine rotor blade (Roy, 1997; Tiwari, 2001)	180
Figure 8.2: Schematic diagram showing general arrangement of coolant flow through turbine blade with film cooling mechanism (1: coolant air inlet, 2: film cooling passage inlet, 3: cooling air exit and 3': film cooling hole exit) (Roy, 1997; Tiwari, 2001).....	180
Figure 8.3: Welded beam design (Deb, 2001).....	181
Figure 8.4: Manufacturing layout design problem (Brintrup, 2007).....	182
Figure 8.5: Results from NSGA-II using two objectives [Plane =(a), Ribbed=(b) and Pedestal=(c)].....	193
Figure 8.6: Results from NSGA-II using four objectives for plane geometry	196
Figure 8.7: Results from NSGA-II using four objectives for ribbed geometry.....	197
Figure 8.8: Results from NSGA-II using four objectives for pedestal geometry.....	198
Figure 8.9: Master-worker representation	200
Figure 8.10: Results of optimisation of welded beam problem using NSGA-II.....	203
Figure 8.11: Implementation interface for rating the designs. (At the top left, when the buttons 0 to 9 are clicked, the values are saved in a file as the objective evaluations; at the top right, the designs are chosen to show new animations of each design).....	205

Figure 8.12: Quantitative fitness in manufacturing plant layout design using interactive NSGA-II	209
Figure 8.13: Qualitative fitness in manufacturing plant layout design using interactive NSGA-II	210
Figure 8.14: Average quantitative values against average qualitative values of manufacturing plant layout model (Gen=Generation).....	210
Figure 8.15: Two different designs obtained for the manufacturing plant layout model (Figure 8.15 (a) shows Store and Office too small and had poor rating of 3; Figure 8.15 (b) shows Store and Office having reasonable sizes and had good rating of 6).....	211
Figure 8.16: Average quantitative values (cost/units) with number of generations for floor planning model (Gen = Generation)	212
Figure 8.17: Average qualitative values (ratings) with number of generations for floor planning model (Gen = Generation).....	212
Figure 8.18: Average qualitative values and average quantitative values for floor planning model (Gen = Generation).....	213
Figure 8.19: Time effect of running the problems on single node and on multiple nodes	215

List of Tables

Table 1.1: Thesis structure.....	12
Table 2.1: Key research papers reviewed based on application areas.....	16
Table 2.2: Some grid projects based on generational classification (sources: Hey and Trefethen, 2002; Buyya, 2002; Berman <i>et al.</i> , 2003).....	31
Table 2.3: PortTypes for Basic Services (Berman <i>et al.</i> , 2003).....	53
Table 2.4: Service Data Elements (SDEs) (Berman <i>et al.</i> , 2003).	53
Table 4.1: Companies/institutions visited during industry survey and criteria for selection.....	75
Table 4.2: Academic and industry opinions on grid features.....	93
Table 8.1: Floor planning upper/lower bounds constraints (Sushil, 1993).....	183
Table 8.2: Common symbols in GTBCSM.....	187
Table 8.3: Constants or design parameters for GTBCSM.....	188
Table 8.4: GTBCSM programming procedure (Sources: Roy, 1997; Tiwari, 2001)...	190
Table 8.5: Variable bounds used to obtain the results in Figures 8.5.....	199
Table 8.6: Welded beam input parameters.....	203
Table 8.7: Quantitative problem parameters for the manufacturing layout design (Brintrup, 2007).	205
Table 8.8: Variable dimensions for floor planning model.....	206
Table 8.9: Average time comparisons for designs generated between one and three nodes.....	220

Normenclature

Abbreviation	Meaning
CA	Certificate Authority
DAME	Distributed Aircraft Maintenance Environment
DECGrid	Decision Engineering Centre Grid
EC	Evolutionary Computing
EDO	Engineering Design Optimisation
EP	Evolutionary Programming
ES	Evolutionary Strategy
FIPER	Federated Intelligent Product Environment
GA	Genetic Algorithm
GEODISE	Grid-Enabled Optimisation Design Search for Engineers
GIS	Grid Information Service
GP	Genetic Programming
GRAM	Grid Resource Allocation Management
GRIA	Grid Resource for Industrial Application
GridFTP	Grid File Transfer Protocol
GSDL	Grid Services Description Language
GSH	Grid Service Handle
GSI	Grid Security Infrastructure
GSR	Grid Service Reference
GT	Globus Toolkit
I-WAY	Information Wide Area Year
MODO	Multi-Objective Design Optimisation
OGF	Open Grid Forum
OGSA	Open Grid Service Architecture
PSE	Problem Solving Environment
QoD	Quality of Data
QoS	Quality of Services
SDE	Service Data Element
SDSC	San Diego Supercomputing Centre
SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol
SORCER	Service-Oriented Computing Environment
SRB	Storage Resource Broker
UDDI	Universal Description, Discovery and Integration
VO	Virtual Organisation
WebMDS	Web Monitoring and Discovery Service
WSDL	Web Services Description Language
WSRF	Web Services Resource Framework

Chapter 1 - Introduction

This opening chapter presents a brief overview and introductory aspects of grid services for multi-objective design optimisation (MODO) applications and the motivation behind using grid technology for MODO in this research. The chapter describes variables that affect collaboration as well as highlights the scope of the research. It ends with the description of the structure of the thesis.

1.1 Overview of the research

The emerging grid computing technology has attracted the attention of different application areas. Researchers in recent times have already spent some time on developing grid middleware and scheduling systems (Foster and Kesselman, 1999). Perhaps the most visible application areas are the computationally and data intensive simulation of large particle physics computations (Bird *et al.*, 2005), multidisciplinary optimisation (Eres *et al.*, 2004), bioinformatics data simulations (Wolstencroft *et al.*, 2007), simulation of oil and gas exploration seismic data (Kurc *et al.*, 2005) and Monte Carlo simulation of commercial/business data (Tezuka *et al.*, 2005). Various grid projects have been sponsored by governments and companies to develop prototypes that will tackle the computational needs of these disciplines (Hey and Trefethen, 2003).

Before the advent of grid computing, computation and data operations were performed on different architectures based on how computational and data intensive the problems are. This motivated Flynn (1972) to propose 4 different architectures for processing data and performing computations. The first is known as the single instruction single data (SISD). This allows data to be processed using sequential instructions. This architecture is good for simple optimisation problems or problems that cannot be parallelised and can only produce single output at a time. This architecture suits classical algorithms for optimisation. Early mainframes were based on SISD architecture. The second architecture is single instruction multiple data (SIMD). This uses multiple data streams to perform single instruction. This applies to problems that can be parallelised. The third architecture is known as multiple instructions single data (MISD). This architecture performs multiple instructions on

single data. This is not a good architecture for performing computations or data processing, but for monitoring faults tolerance. This can be used to monitor a deviation in optimisation results. The fourth architecture is called multiple instructions multiple data (MIMD). This architecture allows multiple processing to be performed on multiple data. This is a truly parallelised system. This is the foundation of distributed computing.

Distributed computing or high performance computing (HPC) is widely used by researchers and companies for computation and data processing. HPC architecture works on a centralised system that provides efficient parallel processing of deterministic multi-objective optimisation but does not provide the robust distributed administrative domain capability for sharing computational resources efficiently for stochastic search algorithms for simulation and optimisation (Nebro *et al.*, 2007). This research will concentrate on grid services in the area of multi-objective optimisation. The grid however incorporates HPC as a subset of its computational resources that can be provided as a service to users along side other computational resources under multiple administrative domains that use both deterministic and stochastic search algorithms (Baker *et al.*, 2002).

High-level (application) grid users are more concerned with grid services that provide easy to use features such as seamless access to the services, graphical user interfaces for job submissions and intuitive process of managing the jobs to completion. Such grid products are termed as problem solving environments (PSE). This research's purpose is to provide such PSE for multi-objective design optimisation (MODO) experts to work in distributed cooperation as they share technical design data, information, computational resources and knowledge. Related PSEs that have attempted to solve problems using grid platforms are GEODISE (Grid-Enabled Optimisation Design Search for Engineers), DAME (Distributed Aircraft Maintenance Environment), FIPER (Federated Intelligent Product Environment), SORCER (Service-Oriented Computing Environment) and Gridbus (Eres *et al.*, 2004; Austin *et al.*, 2005; Sobolewski and Kolonay, 2006; and Buyya, 2002). These PSEs will be discussed in more detail in the next chapter. These PSEs provide the functionalities for component representations of processes in the programmers' specification document expressed as eXtensible Mark-up Language (XML) which can

only be read and understood by the programmers and only deals with the quantitative aspect of problems. This research aims to improve on this by providing both the functional component service specification document that designers can use to implement mathematical models (quantitative) as MODO services as well as taking into account qualitative functionality for interactively visualising designs and rating them on a scale based on the experts' judgement of how the design satisfies certain requirements. In addition, the novelty of the research is the inclusion of a structured approach and guideline for implementing these services in the proposed framework. This process will be discussed in Chapter 6. This approach allows MODO experts to see for themselves which type of grid services is best suited for their application as these issues will be part of the service level agreement. This gives the research a perspective of the grid as a utility meant to serve end users by putting their satisfaction of the service in the minds of programmers as contained in the documents. It is important to mention here that the initial concept of the grid coined from electricity grid is to provide computational resources as services from a utility point of view (Buyya, 2002).

Having given an overview of the research as above, the main focus behind this chapter will be stated. The purpose of this chapter is to:

- *Give an overview of the research*
- *Introduce grid services and multi-objective design optimisation*
- *Highlight the motivational features of grid services for MODO applications in this research*
- *State the scope of the research*
- *Summarise the structure of the thesis*

1.2 Introduction to grid services

Grid services emerged from the concept of web services which have been the fundamental tool for business-to-business and enterprise integration of transactions, collaboration and communications among business partners and customers in today's world of electronic commerce (e-commerce) and globalisation. The World Wide Web Consortium (W3C) group defines web service as a software application which is identified by a Universal Resource Identifier (URI) whose interfaces and bindings are capable of being defined, described and discovered as Extensible Markup Language (XML) artifacts (Ferris and Farrell, 2003). In essence, it is a software application that

when invoked provides its functionalities to the user as a set of services. A grid service extends this capability to include ‘stateful’ implementation of services using Grid Services Description Language (GSDL) and Web Services Resource Framework (WSRF) (Foster, 2005). By ‘stateful’ implementation, it means that grid services, unlike web services, can be discovered, invoked and manipulated dynamically through their values, types and unique identities even after their life time using the Grid Service Reference (GSR) and Grid Service Handle (GSH) properties (Berman *et al.*, 2003). GSDL is the equivalence of Web Service Description Language (WSDL) for grid services which uses XML syntax to present the functionalities and interfaces to other services and users. The WSRF provides the capability for web services and grid services to work on the same platform, providing convergence for the two. GSR and GSH provide unique identification for each instance of grid service invoked to be used without interference by another instance. These concepts will be explained in chapter 2.

The Open Grid Services Architecture (OGSA) provides a platform called the Service-Oriented Architecture (SOA) for grid resources to be published as services. SOA describes the policies, practices and frameworks that enable application functionalities to be provided (by service providers) and consumed as sets of services at granularity relevant to the end service consumer (requestor) (Spratt and Wilkes, 2004). This feature is used by grid middleware such as Globus to partition different levels for different grid resources. In this research, communication among different nodes are managed at the network and communication level while the application interfaces for optimisation inputs are run at the application level of Globus Toolkit. Three main components of the SOA as can be seen in Figure 1.1 below are publish (provision of) services, find (discovery of) services and subscribe (consumption of) services.

Figure 1.1 shows that the service provider publishes resources as services which can be subscribed by a requestor (user). The subscription is possible using the monitoring and discovery service interface. Multiple providers can publish different resources (Resource 1, Resource 2, Resource n) for different users to use. One challenge in SOA is resource and service discovery for multiple providers and users (Foster *et al.*, 2001). The service subscription is initiated by the user by invoking a notification service. The notification service notifies the provider and a service or resource is

made available to the user. The provider acknowledges receipt of a request for a service.

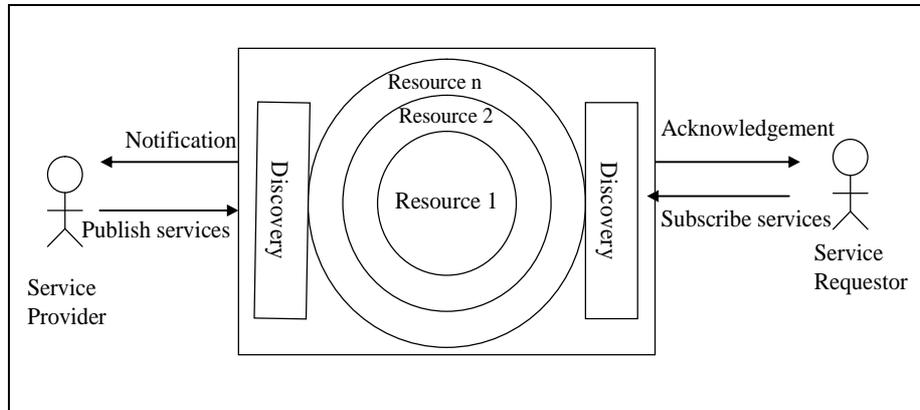


Figure 1.1: Description of interactions in Service-Oriented Architecture

1.3 Introduction to multi-objective design optimisation (MODO)

Engineering and scientific real-world products increasingly need improvement. This improvement may be a combination of two or more improved performance, cost, durability, manufacturability, portability or usability. To achieve a combination of these objectives based on different users' requirements, some trade-off or compromise is required during the optimisation phase. This compromise means considering multiple objectives against certain constraints to get an improved (optimised) design that satisfies the requirements stated by the user. The process of considering many objectives in the face of constraints to improve a scientific or engineering product through the selection of an optimum solution from many solutions during the design phase is known as multi-objective design optimisation (MODO) (Sasaki *et al.*, 2006).

Different methods and techniques are used in MODO. These methods are broadly classified as classical or evolutionary methods. Classical method uses manual computational techniques such as generating methods and preference-based methods to reduce a multi-objective problem to a single objective problem while evolutionary method uses genetic algorithms (GAs) to produce multiple solutions in a single simulation run (Deb, 2001).

MODO experts use the concept of dominated and non-dominated solutions to justify the reason for selecting particular solutions over others. A solution is said to dominate another if the solution is better when different objectives are considered and never worst for any of the objectives. Two solutions are considered non-dominated to each other if one performs better under certain objectives and worst under different set of objectives under consideration. In this case, none of the solutions is better than the other if all the objectives are considered. The set of solutions that are not dominated by any member of the solution set is known as the Pareto-optimal solution set. The curve that joins the Pareto-optimal solution set is known as the Pareto front. The main goal in any multi-objective optimisation problem using GA is to obtain the Pareto front.

1.4 Motivation for grid services in MODO applications

MODO problems are computationally expensive, data intensive, multidisciplinary and collaborative in nature. Because of these challenges, providing computational resources to MODO experts as distributed services make the grid an attractive problem solving environment. The motivational factors for developing grid services for MODO applications in this research are discussed below.

1.4.1 Computational motivation

Modern processing environments that consist of large collections of workstations interconnected by high capacity network raise the following challenging question: can we satisfy the needs of users who need extra capacity without lowering the quality of service experienced by the owners of under utilized workstations? . . . The Condor scheduling system is our answer to this question.

– (Litzkow *et al.*, 1988).

In the days of mainframe computers, efficient utilisation of computational power was guaranteed in centralised distributed systems. In other words, there was no wastage of computational power, though the centralised system is inefficient in handling computationally intensive jobs. Today's personal computing and client/server technology has provided a more efficient system, but with the resulting waste in idle personal computing power. This is to say that there is enough computational power that is not put to use to solve computationally intensive jobs such as the MODO

applications. MODO applications usually result in repeatedly long running solution processes even when using state-of-the-art parallel/distributed computing facility (Grauer *et al.*, 2004). Grid computing provides idle computational power (processors and hard disks) as services for users to subscribe. Even within the same organisation, computationally intensive jobs can be flocked (migrated) for processing to idle systems using the concept of ‘cycle-stealing’. The grid resource scheduler responsible for cycle-stealing is known as Condor-G. With this concept, companies involved in design optimisation, computational fluid dynamics (CFD) and finite element analysis (FEA) may not need to buy expensive supercomputers, but can harness the computational synergy of their personal computers to overcome certain computational challenges.

1.4.2 Data motivation

Terabytes and petabytes of data are generated from computations and simulations of MODO applications. These datasets are used and shared by multidisciplinary experts during design. The challenges in sharing and managing these huge datasets in conventional distributed systems are storage, transferring, allocation, access and retrieval (Wang *et al.*, 2007). The Globus Toolkit (GT) which is the grid middleware used in this research has built-in services such as Grid File Transfer Protocol (GridFTP) which extends the capabilities of FTP to speedup data transfer to far away remote grid nodes (clients) and Grid Resource Allocation Manager (GRAM) manages the data allocation, access and retrieval dynamically. Storage services for data/metadata can be provided by some of the nodes within the grid. A fundamental challenge which data grid can handle is the creation, registration, location and management of dataset replicas to maintain data integrity and consistency in the entire grid (Allcock *et al.*, 2002). Grid problem solving environments (PSEs) are used in the search and exploration of designs at the centres of expertise in engineering design optimisation (Parmee *et al.*, 2005). This research takes advantage of this grid feature to configure some nodes of the grid for building mathematical model and this model can be transferred and used for optimisation by other nodes whose results can be used by some other nodes. The GridFTP and GRAM allow these quick transfers, storage and data management.

1.4.3 Multidisciplinary and collaboration motivation

MODO experts collaborate in multidisciplinary fields to come up with optimum designs. Grid environment has been used to enable multidisciplinary experts to collaborate on a service-oriented architecture to enhance concurrent engineering design optimisation using Federated Intelligent Problem Environment (FIPER) and Service-Oriented Computing Environment (SORCER) grid environments (Sobolewski and Kolonay, 2006). In this research, a collaboration functionality called collaboratory is implemented to enable MODO experts to collaborate for mathematical model building. This work is different from FIPER and SORCER by including a step-by-step process that builds a model. These steps are derived from literature and interviews with design experts during industry survey. The model building process is provided as a service. Figure 1.2 is the collaborative process diagram of building the mathematical model and getting the qualitative fitness from design experts for non-dominated sorting algorithm two (NSGA-II) to perform crossover, mutation and ranking operations. Detail of each step is explained in section 5.3 of chapter 5.

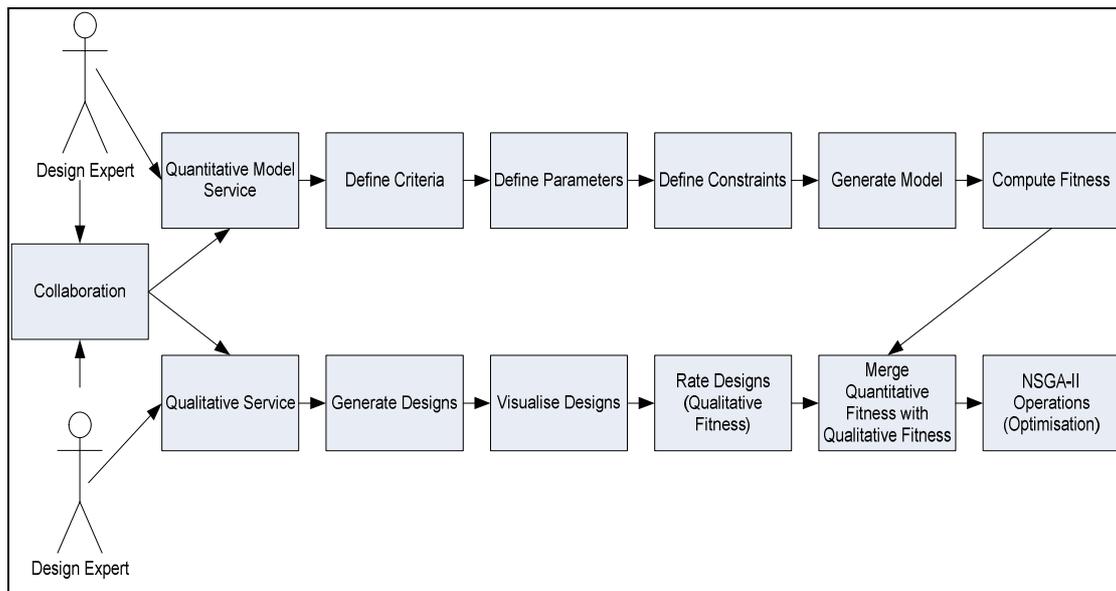


Figure 1.2: Process of mathematical model building, collaboration and optimisation services

1.5 Key variables that affect collaboration

To be able to design, develop and implement problem solving environments for collaboration, it is crucial to identify the factors that affect such collaborations. Virtual Organisation (VO) is a collection of users that are connected together and share network resources to accomplish a common goal. VOs are emerging as a result

of collaboration within grid environments and the key variables that play important roles in collaboration are identified and discussed in this research as below.

1.5.1 Security

Collaboration will not take place when participants are not assured of the safety of their resources such as data, information, networks, instruments and business rules. This informed the seriousness with which the Grid Security Infrastructure (GSI) is implemented in the Globus Toolkit. GSI is the feature which authenticates and authorise users to have access to grid resources. For a user to use the grid, an issuing body called Certificate Authority (CA) needs to issue the user a certificate. This certificate contains the user identification and passphrase which are encrypted. The proxy certificate issued by CA ensures strict compliance with policy guiding the collaboration contract. File transfers are encrypted through the Grid File Transfer Protocol (GridFTP) which is used for transfer of files and data. Secured remote transfer of large datasets is done through secured shell login facility known as GSISSH (GSI Secured Shell). Despite this security features, industrial grid application users are still sceptical about the safety of their data. This is because the grid concept allows distributed and third party users to give access to resources without the knowledge of the original owners of the resources through a proxy authorisation system (Foster *et al.*, 1998a).

1.5.2 Autonomy

Collaborating organisations want to have control over their grid resources as they allow others in the grid. When and who accesses the grid should be guided by the policy agreed upon by grid providers and requestors. More importantly, companies would want to maintain autonomy over their administrative domain to preserve their organisation structures which includes identities of users' certificates, location and contents of key data repositories (Foster *et al.*, 1999). Grid Information Service (GIS) provides persistent service for collaborators to access different resources from different locations while GSI provides security checks to ensure autonomy. During the industry survey in this research, one of the companies said that the need to have autonomy on their data and information prompted them to develop an in-house middleware called Grid Resource for Industrial Application (GRIA). The GRIA

project aims at preserving the autonomy of grid collaborators while providing virtualised grid resources (SurrIDGE and Taylor, 2005).

1.5.3 Performance

Different grid models look at performance and optimum utilisation of resources as key to encouraging the formation of VOs. Performance includes speed of processing data, bandwidth and network responses. Condor and Globus resource management system called Globus Resource Allocation Management (GRAM) are expected to improve computational performance through cycle stealing as well as reduce ‘waste’ as desktops stay idle most of the times in companies. In this research, the concept of cycle-stealing is used to flock jobs to idle systems. The question is-will other users of the grid allow their systems to be used when they are idle? This is why it is important to identify the variables that affect such collaborations.

1.5.4 Hosting environment

Grid services are created and maintained by hosting environments (HE). These HEs determine the life, quality and quantity of services published. HEs also define the implementation of programming models, programming languages, development tools and stipulate how grid services meet their obligations to the community of grid users. This complements the activities of Open Grid Services Architecture (OGSA) as OGSA does not put restrictions on service requirements. The Component Object Resource Broker Architecture (CORBA) in java programming language ensures cross implementation of different languages such as C, C++, Perl, Fortran and VB.NET on the grid (Foster *et al.*, 2003).

1.5.5 Quality of service

Participating grid users define quality of services. HEs instantiate services and ensure service requirements are adhered to. This includes quality of service (QoS). Quality of service includes availability, reliability, performance and many more. OGSA (Open Grid service Architecture) has standard interface implementation rules for participating grid users to ensure smooth exchange of quality of services. In this research, the recommendation to include the process of service implementation is to provide for service level agreements that will take care of QoS issues. This is contained in the service specification document preposed for this research. This part of the shortcomings is identified in most PSEs.

1.5.6 Reliability

Grid platforms need to be stable for users to collaborate effectively. Fault tolerance services use portTypes (interfaces) such as NotificationSource and NotificationSink to notify grid users or services of problems (Berman *et al.*, 2003). The Globus resource manager (Globus Resource Allocation Management) plays a major role in resource allocation. GRAM uses the gatekeeper component to authenticate users and the jobmanager feature manages the the resource allocations to distributed users. Each node in the grid is expected to have its GRAM. Reliability is still an open issue in grid research (Andrieux *et al.*, 2003).

1.5.7 Scalability and interoperability

Implementation of grid services using the plumbing interface method ensures scalable grid infrastructures. The plumbing interface method is the concept which grid developers adopt to follow the conventional standards specified in OGSA (Open Grid Service Architecture) and WSRF (Web Services Resource Framework). For as long as grid participants follow the OGSA and WSRF conventions of implementing grid interfaces (portTypes), participation in grid of any size will not present scalability and interoperability problem. Middleware such as Globus implements protocols such tha ensures scalable collaboration. The layered architecture of the grid is intended to make it scalable and interoperable among heterogeneous resources (Foster *et al.*, 2001).

1.5.8 Transparency

Grid requires network neutrality, location neutrality and communication neutrality to efficiently allow collaboration within a VO. This means that no matter what network a grid user is using, smooth communication is guaranteed within grid environment. Middleware and problem solving environments used the OGSA and WSRF standards to ensure platform and location transparency.

1.6 Scope of the research

This research is about developing a grid methodology for MODO application in science and engineering. This research is limited to the following aspects of grid service application in MODO:

- MODO service specification methodology
- MODO service architecture development

- MODO service interface implementation
- Mathematical model building service
- Optimisation service
- Collaboration functionality
- Computational synergy

This scope is adopted based on the duration of the research and the essential requirements for MODO. The essential requirements are the concept, methodology and infrastructure that creates enabling environment for designers to perform MODO.

1.7 Structure of the Thesis

The thesis consists of 9 chapters. Table 1.1 below summarises the structure of the thesis.

Table 1.1: Thesis structure

Chapter	Title	Summary
1	Introduction	Chapter 1 is a brief overview of grid computing, grid services, multi-objective design optimisation and motivation of using grid services for MODO. The scope and structure of the research are also part of chapter 1.
2	Review of literature	Chapter 2 is the review of literature. This consists of related problem solving environments, generations of grid services and classification of grid application areas. The gaps in literature are identified from the classification of literature.
3	Research aim, objectives and methodology	Chapter 3 describes the aim, objectives and methodology. The steps in the methodology are literature review, industry survey, analysis and design of MODO grid service, development, case study and validation.
4	Grid in industry	Chapter 4 presents the findings from industry survey conducted in aerospace, oil and gas, software companies and the grid research community using interviews and questionnaires. These findings were compared with the findings in literature and gaps between research and industrial applications of grid were obtained. The aim of the industry survey is to give the research an industrial context.

Chapter	Title	Summary
5	Grid service specifications and design for MODO	Chapter 5 analyses the service specifications of grid applications in general and then provides specific service specifications for Multi-objective design optimisation (MODO) services and how to expose their functionalities to users. These specifications are used in the design of the Decision Engineering Centre Grid (DECGrid) prototype using unified modelling language (UML).
6	Service framework and architecture of DECGrid	Chapter 6 uses the service specifications in chapter 5 to come up with a generic framework and architecture for the DECGrid prototype.
7	Implementation of DECGrid and MODO services	Chapter 7 is the implementation of DECGrid. Linux operating system, Globus Toolkit middleware, Condor scheduler and PostgreSQL database server are used as software platforms. 8 grid nodes are used as the hardware platform.
8	Validation using case studies	Chapter 8 is the validation of DECGrid using 3 case studies. The case studies are optimisation of gas turbine blade cooling system (aerospace), welded beam problem (engineering) and design of a manufacturing plant layout/floor planning (architecture). Experts are used to perform hands-on validation of the prototype using different real-life scenarios. Questionnaires are given to the experts to fill after the hands-on session. The results obtained are presented and discussed with comparison to the results in literature.
9	Discussion and conclusions	Chapter 9 ends with a discussion on the entire research and makes suggestions for further research.

1.8 Summary

Chapter 1 covers introductory research aspects of grid services in multi-objective design optimisation. The motivation, scope and research structure were stated. This now prepares the research to focus on literature review to gain deeper insight into the research.

Chapter 2 - Review of Literature

This chapter reviews past, current and future research findings on grid computing. It also reviews literature on multi-objective design optimisation, service-oriented architectures and related problem solving environments and how grid technology is associated with them. The literature review led to the classification of the evolution of grid computing and looked at the future trend in grid research. From this classification, gaps in the research area were identified and this formed the basis of the research aim and objectives in the next chapter.

2.1 Introduction

The early concept of grid computing is simply to link supercomputing centres so that a computational synergy may be obtained to overcome computationally expensive jobs that could not have been left to a single supercomputing centre. The Information Wide Area Year (I-WAY) project is one of the earliest projects to bring together researchers to demonstrate this computational grid computing synergy by linking 17 supercomputing centres (Stevens *et al.*, 1997). This concept of linking supercomputing centres is known as meta-computing. This notion of meta-computing was later transformed into computational grid, a name coined from the electricity grid which transformed the United States and indeed the world over in 1910 (Foster and Kesselman, 1999). The notion of computational grid triggers the idea of integrating the grid with existing technologies such as the web and virtual reality computing (Stevens *et al.*, 1997). The middleware concept was born out of the need for this integration process. As time goes on, the initial concept of the grid keeps changing. This is the basis upon which the researcher considers it important to classify the evolutionary trend of grid computing as part of the literature review so that the idea of service-oriented grid for MODO applications can be appreciated better. At the time of this research, commercial and business giants Amazon, Google and International Business Machines (IBM) are coining the same concept into what is referred to as cloud computing (Hayes, 2008).

MODO is one of the application areas that have witnessed interest in using grid technology. This perhaps is because MODO uses algorithms that are computationally

and data intensive in nature. The literature review also looked at MODO and its characteristics that suit the grid as a problem solver. To do this, some related grid projects were identified and discussed. MODO approaches such as classical and evolutionary methods were discussed to identify which approach best suits the grid technology. To this end, it is important to state the purpose this chapter intends to achieve.

The purpose of this chapter is to:

- *Review literatures in grid computing and MODO.*
- *Classify the evolution of grid computing.*
- *Review some related grid PSEs, middleware and scheduling systems.*
- *Identify some related key grid projects.*
- *Identify challenges of implementing grid solutions for MODO applications.*
- *Choose an MODO method that is suitable for this research.*
- *Identify the tools and techniques used for implementation of grid services.*
- *Identify the gap in literature.*

2.2 Grid computing

The state of play in the metamorphosis of grid computing is similar to the evolution of programming languages in the early 1960s. From a purely mathematical and scientific tool for solving complex scientific problems by mathematicians and scientists to a more friendly, human-like procedural and later object-oriented (OO), aspect-oriented (AO), service-oriented (SO) and component-oriented (CO) languages for solving mathematical, scientific, commercial, political, and almost every day human problems that will bring automation and efficiency. Just like programming languages which started their evolution with the first generation languages to the present sixth generation languages, the grid is now in its fourth generation of evolution and development. Grid computing is a computer science discipline that promises to address distributed large-scale and multidisciplinary infrastructural issues with its reliable, inexpensive, secured, pervasive, dependable, and coordinated resources (Foster *et al.*, 1999). In its evolution, the grid has evolved from computational and data intensive platform for solving large-scale scientific problems to a service-oriented problem solving environment (SO-PSE) for solving multidisciplinary problems that may have multi-objective criteria. As stated in section 2.1, the research

will review the trend of evolution of the grid so that the perspective in which the title of the research is conceived can be understood. The classification looks at the application areas and their requirements for using the grid and also the challenges that often lead to successive levels of evolution.

2.3 Generations of grid computing

In its evolution, the grid has passed through research stages that can be conveniently classified as first, second, third (De Roure *et al.*, 2003), fourth and future generations of grid computing. The first generation deals with computational and data intensive applications in science and engineering. The second generation deals with protocols and middleware to overcome interoperability problems among grid users. The third generation is concerned with problem solving environments (PSE) and the fourth generation is concerned with service-oriented architectures (SOA) and economic models that can transform the grid into a full-fledged utility computing platform (Abramson *et al.*, 2002). The future generation addresses issues relating to semantic grid, knowledge grid, autonomous grid and the emerging concept of cloud computing ((Ramakrishnan, 2008), (Hayes, 2008) and (Liu and Orban, 2008)).

Table 2.1: Key research papers reviewed based on application areas

Application Areas	Computationally Intensive	Data Intensive	Knowledge/ Semantic	Visualisation	Service-Oriented
Authors	Foster and Kesselman (1999), Foster <i>et al.</i> (2001), Goux <i>et al.</i> (2001), Goodyer <i>et al.</i> (2005), Cao <i>et al.</i> (2003), Cox <i>et al.</i> (2001), De Roure <i>et al.</i> (2003), Sasaki <i>et al.</i> (2006), Fox <i>et al.</i> (2002), Defanti <i>et al.</i> (1996)	Venugopal, <i>et al.</i> (2004), Wan <i>et al.</i> (2003), Wang <i>et al.</i> (2007), Foster <i>et al.</i> (2002b), Fox (2003), Rajasekar <i>et al.</i> (2002), Cannataro <i>et al.</i> (2003a), Bell <i>et al.</i> (2003), Antonioletti <i>et al.</i> (2003), Stockinger (2002), Deelman <i>et al.</i> (2004) Deelman <i>et al.</i> (2002), Parashar <i>et al.</i> (2005a), Dullmann <i>et al.</i> (2001), Rajasekar <i>et al.</i> (2003), Vazhkudai and Schopf (2002), Kosar and Livny (2004), Antonioletti <i>et al.</i> (2005), Barbera <i>et al.</i> (2003), Aloisio <i>et al.</i> (2004)	Gil <i>et al.</i> (2004), Hau <i>et al.</i> (2003), Schwidder <i>et al.</i> (2005), Fox (2003), Chen <i>et al.</i> (2004), Zhuge (2004), Canntaro and Talia (2004), Goble and De Roure (2002), De Roure <i>et al.</i> (2005), Pouchard <i>et al.</i> 2003), Zhuge and Liu (2005), Ghanem <i>et al.</i> (2002), Cannataro <i>et al.</i> (2001), Li and Lu (2004), Schikuta and Weishaupl (2004), Cannataro and Talia (2003a) Cannataro and Talia (2003b), Blythe <i>et al.</i> (2003), Boose (1989)	Foster <i>et al.</i> (1999), Norton and Rockwood (2003), Bethel and Shalf (2003), Kranzlmuller <i>et al.</i> (2003), Brodlie <i>et al.</i> (2004a), Shalf <i>et a</i> (2003), Brodlie <i>et e l.</i> (2004b)	Abdelzaher and Shin (1998), Pearlman <i>et al.</i> (2002), Venugopal <i>et al.</i> (2004), Foster <i>et al.</i> (2002a), Barmouta and Buyya (2003), Kacsuk <i>et al.</i> (2004), Hwang and Aravamudham (2004), Stevens <i>et al.</i> (1997), Talia (2002), Welch <i>et al.</i> (2003), Papazoglou (2003), Andreozzi <i>et al.</i> (2003), Goel and Sobolewski (2003)

Table 2.1 is a list of some authors that have done research in different grid application areas. These key papers were reviewed during this study and authors that worked under computational intensive, data intensive, knowledge/semantic, visualisation and service-oriented application areas are listed accordingly. This arrangement from computational to service-oriented grid shows the trend of application areas. For example, Foster and Kesselman (1999) are the pioneering researchers in the early days of meta-computing during which computation and data capabilities of the grid were more important. Later uses of the grid were discovered and this gave rise to the different applications of grid. Some papers discussed more than one application areas and such papers appear more than once in different application areas. Almost all these application areas are common features of MODO. MODO is computationally expensive, data intensive, knowledge driven and requires visualisation for decision making (Chervenak *et al.*, 2000). This is one of the reasons of choosing grid services for MODO in this research.

2.3.1 First generation of grid computing

As has been mentioned previously, the first generation of grid computing is basically linking of different supercomputing resources to produce computational synergy among users at different sites to enhance multidisciplinary cooperation in research. The components of the first generation grid are discussed below.

2.3.1.1 Computational grid

In 1993, the National Science Foundation (NSF), National Computational Science Alliance (NCSA) in Illinois and the San Diego Supercomputer Centre (SDSC) in California brought together a number of supercomputing centres together to share computing power. This project proved to have a great synergistic effect. The result of this was the use of redundant computing cycle-time which resulted in faster computation, better computational results, cheaper maintenance cost and effective throughput. This formed the first generation of grid computing application which is popularly referred to as metacomputing (Berman *et al.*, 2003). The whole idea of metacomputing is to harness the computing power that was ‘littered’ around at different geographical locations within the United States supercomputing stations. This is because none of them (supercomputing stations) was able to single-handedly solve the scientific computations that involve large computational fluid dynamics

(CFD) and other astrophysics computations. It was purely for scientific purpose among researchers and the academia. The conventional way companies perform engineering design optimisation is by using the local computing power they have at their disposal. The use of high performance supercomputers can be very expensive. The introduction of computational grid is a significant milestone that addressed this problem. With grids, the optimisation processes can be sub-divided into smaller tasks and launched separately at different grid centres perhaps owned and maintained by different (erstwhile supercomputing centres) organisations. In this way, the use of computational grid, either across a single large enterprise or between many enterprises provides a great deal of opportunities for cost-effective access and usage of large-scale computational resources with very good throughput and results that ultimately yield the desired and improved design (Goodyer *et al.*, 2005). The grid provides a convenient platform for handling the computationally and data intensive aspect of MODO problems (Luna *et al.*, 2006).

Figure 2.1 shows number of research papers reviewed in different application areas. Majority (38) of the papers while discussing other application areas relate them to service-oriented architecture (SOA) of the grid. Thus SOA forms the intersection for grid applications. This review reveals that research is currently concentrated on service-oriented architecture (SOA) to provide computational, data and visualisation as services. Current grid application frameworks use SOA to (1) achieve remote accessibility of computational capabilities and resources (2) communicate using messages and (3) manage metadata (Fox *et al.*, 2005). These core functionalities are required in MODO for computational resource sharing.

2.3.1.2 Data grid

Data grids have evolved to cater for the dual challenges of large datasets and multiple data repositories at distributed locations in data-intensive computing environments (Venugopal *et al.*, 2004). Venugopal *et al.* (2004) described how large communities of researchers around the world are engaged in the analysis, collation, and collection of data generated by scientific instruments and replicated on distributed resources. For example, applications for MODO environments need to have grid-enabled resource brokers that will allow design engineers access to distributed data and computational resources. The resource brokers help to progressively discover, select, and publish

required resources in an intelligent manner. Though the grid enables the aggregation and sharing of resources among design experts, harnessing the power of grids from different nodes and sites of a Grid is still a challenging problem due to the complexity involved in the dynamic creation and heterogeneous composition of resources. The San Diego Supercomputing Centre (SDSC) uses Storage Resource Broker (SRB) as a data grid middleware to integrate various types of data ranging from digital libraries to persistent archives stored in various databases and formats (Wan *et al.*, 2003).

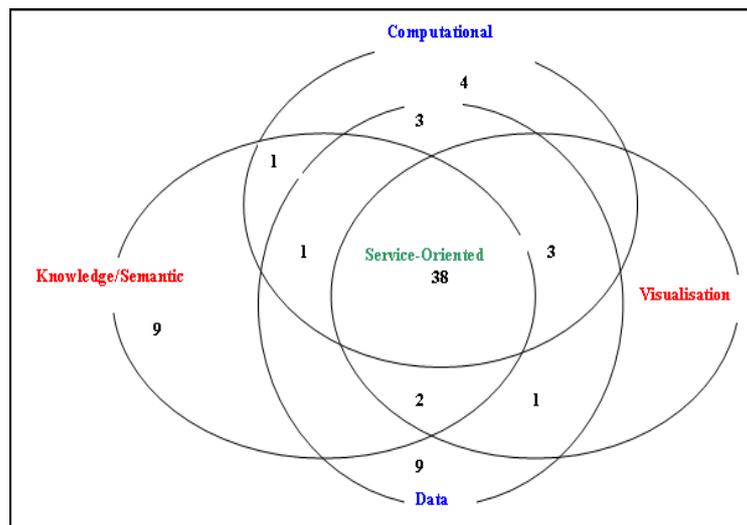


Figure 2.1: Grid application areas and number of research papers

A review of papers according to application areas as shown in Figure 2.2 shows that data application received more attention from the research papers (46) than other application areas. This shows that data occupies an important position in the grid evolution. Visualisation received the least (11) attention. This shows that grid visualisation at the time of this research is still at its infancy. Knowledge/Semantic grid occupies the second position (31) in the chart, trailing behind data grid.

First generation of grid computing (computational and data grids) was faced with some challenges. There was the problem of interoperability among the linked stations because of the heterogeneity of the resources (hardware, software and data formats) each is using. There was also lack of real collaboration and visualisation of results coupled with a lack of knowledge/semantic driven grid that will ensure reuse.

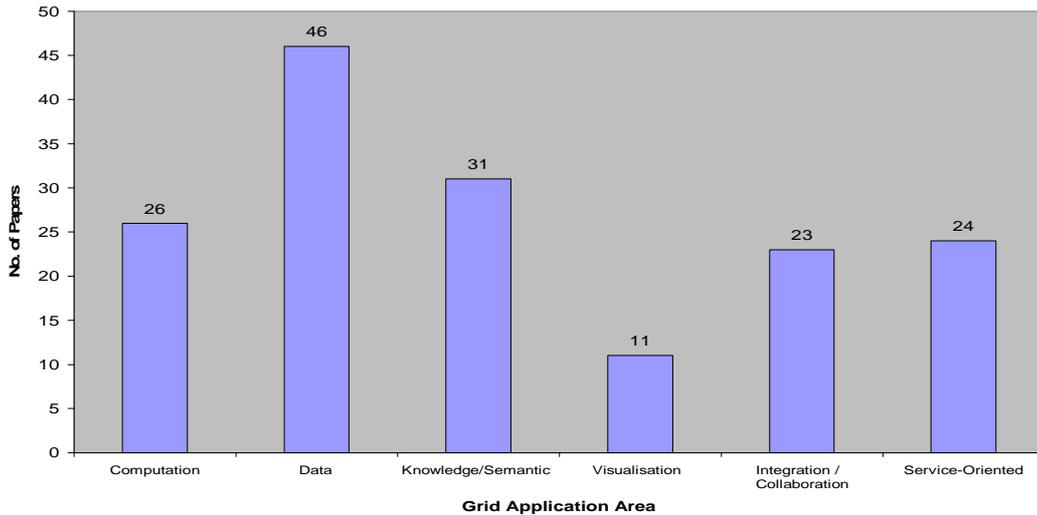


Figure 2.2: Grid Application Areas and Number of Research Papers

This takes the research to the next level-the second generation of grid computing research activities.

2.3.2 Second generation of grid computing

The second generation of grid computing concerns itself with communication protocols and middleware issues to solve interoperability related problems among collaborating supercomputing stations. The main areas of research are reviewed below.

2.3.2.1 Middleware

The immediate problem faced by metacomputing (first generation grid) era was interoperability. Having successfully linked the different supercomputing centres to harness computing power, the heterogeneous components (software and hardware) of the different centres hindered smooth communication. This problem formed the main discussion topic by the Global Grid Forum (GGF) (now Open Grid Forum (OGF)). The middleware concept to ensure interoperability among the supercomputing centres is the first initiative towards the second generation of grid computing. At the OGF meeting in 1993, Professor Ian Foster and Professor Carl Kesselman were charged with the responsibility to head a team to design and develop a robust grid middleware. In 1995, the first tested grid middleware, called the Information Wide Area Year (I-WAY) was demonstrated by 17 supercomputing centres and other computational research laboratories at the supercomputer '95 (SC95) in San Diego. I-WAY later evolved into the most popular grid middleware called the Globus Toolkit (GT)

(Berlich *et al.*, 2005). GT is the *de facto* middleware for grid applications among researchers all over the world. The middleware aims at solving the interoperability problem witnessed in the first generation of Grid testbeds. In this research, one of the validation scenarios is to demonstrate that the prototype (DECGrid) provides an environment for MODO experts to communicate and collaboration. This means that a middleware is essential in ensuring a hitch-free collaboration among distributed MODO experts who may want to share optimisation math model and results. The DECGrid prototype is built on Globus middleware.

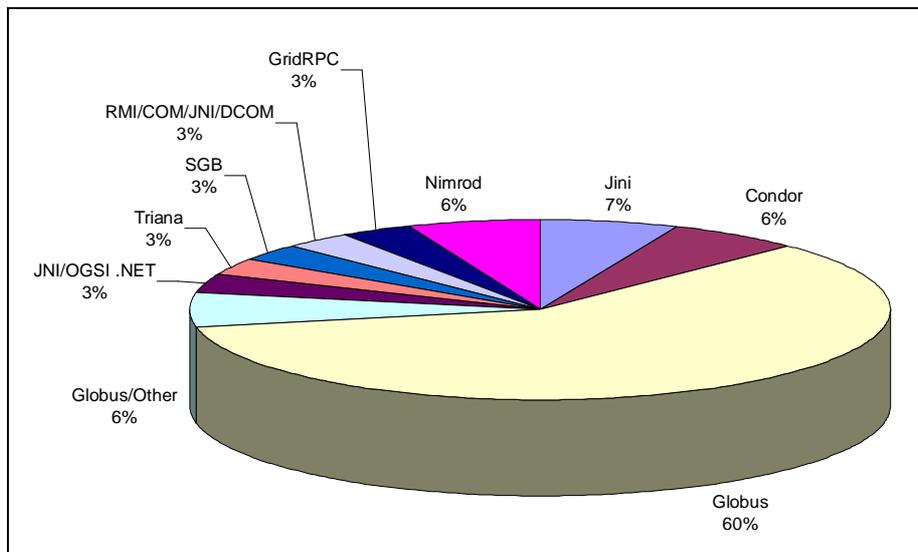


Figure 2.3: Middleware and percentage of papers

Figure 2.3 shows the percentage of research papers reviewed against various middleware used for MODO and related applications. Globus middleware occurred most (60%) in the research papers demonstrating its success among researchers. Globus is sometimes used with other middleware to add more functionality to the user. This is one of the reasons why the researcher chooses to use Globus as a middleware for this research. Jini got 7% and Condor, Globus/Other and Nimrod middleware received 6% each.

2.3.2.2 Visualisation grid

Another challenge the first generation grid-enabled infrastructures faced was the need to obtain real-time results and visualise them to acquire better understanding for decision making. The computational steering of parameters at run time for

engineering design optimisation as the designer visualises the effect is very important. This allows other collaborators in the design process to join in the simulation and to interact with the design system through both visualisation and steering from other sites of the grid nodes (Goodyer *et al.*, 2005). The common thinking among the users is that computing power and resources are needed away from local needs and that applications need to use distributed resources (Goodyer *et al.*, 2005). Goodyer *et al.* (2005) maintained that Grid applications which are computationally intensive and collaborative in terms of the scientific community lead to two important questions. These are (1) how can knowledge and insight be acquired quickly from a grid application that runs on the distributed nodes? (2) How can these results and knowledge be shared among geographically dispersed scientists who might have different backgrounds and expertise? The recent advances in computer visualisation and grid-enabled Virtual Reality (VR) allow designers to interact and manipulate vast amounts of data and metadata to further improve their efficiencies in producing competitive engineering products at affordable costs.

2.3.2.3 Knowledge grid

The sequence of the grid evolution continues with each successful or near successful implementation. Having realised the importance of visualisation among collaborative grid users, there is the need for these collaborators to share and reuse knowledge. A multidisciplinary field such as engineering design optimisation requires that each domain expert shares his or her expertise with others so that there can be a complete learning cycle for more efficient design process. A vast amount of information is generated and stored in digital repositories during optimisation, yet it is often difficult to understand the important and useful information in those massive datasets (Cannataro and Talia, 2003a). The databases holding optimisation information need knowledge discovery agents to guide new optimisation processes.

2.3.3 Third generation of grid computing

Third generation of grid computing addresses the need for service-oriented architecture and problem solving environments.

2.3.3.1 Service-oriented grid computing

Like all infrastructures, users want to see how grid computing can be applied directly to solve their daily needs. For example, infrastructures such as electricity, railroads,

and telephone were successful because of the visible service they provided at their inception as services (Foster and Kesselman, 1999). Users will naturally endorse infrastructures when the benefits are practically visible in the way they do things. This perhaps informed the decision of researchers to develop a grid utility model. Problem solving environments such as FIPER (Federated Intelligent Product Environment), SORCER (Service-Oriented Computing Environment) and GEODISE (Grid-Enabled Optimisation Design Search for Engineering) were developed in line with this vision. Figure 2.4 shows the number of research papers that focused on different service-oriented models. Geodise seems to be the favourite especially within e-Science (e-Science is the European concept of large-scale distributed computational infrastructure for doing science and engineering) optimisation and computational research domain. OGSA (7 papers) and FIPER (4 papers) are also popular among researchers within service-oriented and concurrent optimisation domains.

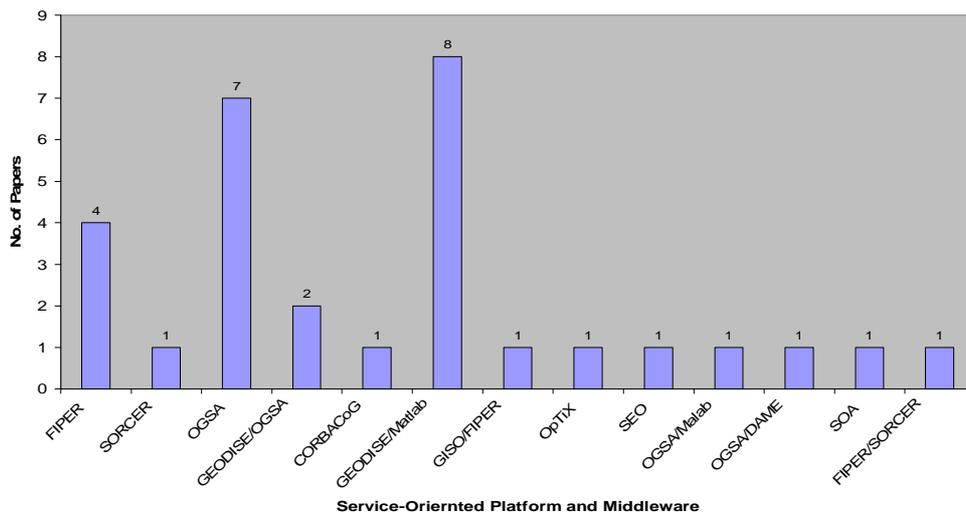


Figure 2.4: Service-oriented Models and Number of Papers

Service-oriented grid-enabled computing has the robust capabilities to enhance collaboration in an extended manner that has never been possible (Foster, 2005). For example, in MODO where computing cycle, data storage, optimisation codes, multidisciplinary knowledge, and expertise are distributed across geographical boundaries around the world, cooperative organisations can agree on service (supply and demand) policies and register on a grid in which published resources are available for usage by all (Song *et al.*, 2004). These services can be accessed through the web or grid services. Figure 2.5 is the process of Web and Grid convergence in technology

and application areas. The convergence point is facilitated by the Web Services Resource Framework (WSRF). The convergence shows the trend from I-WAY (Information Wide Area Year), Globus versions (GT1 to GT4) to OGSA (Open Grid Services Architecture) and GSDL (Grid Services Description Language) on the Grid site. On the Web convergence site, the convergence begins from HTTP (Hypertext Transfer Protocol) to WSDL (Web Services Description Language), SOAP (Simple Object Access Protocol) and UDDI (Universal Description, Discovery and Integration). OGSA consists of platforms and interfaces that provide the infrastructure for protocols such as SOAP and WSRF to function. HTTP is a stateless application level protocol that can support collaborative and distributed representation of data in systems. UDDI is a platform-independent protocol that is based on XML and is used by organisations to make themselves available and be discovered by other services on the internet or grid systems.

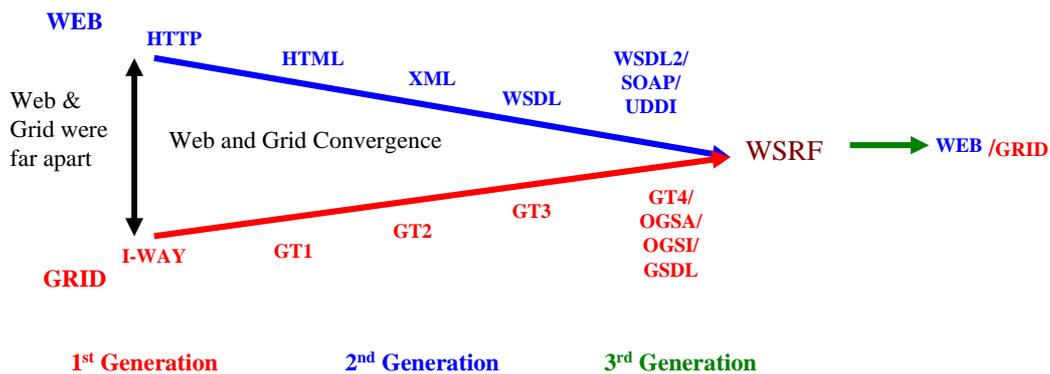


Figure 2.5: Web and Grid Convergence Process as the Generations of Grid Mature (Source: Foster *et al.*, 2001)

2.3.3.2 Grid computing-enabled problem solving environments

Having incorporated service-oriented concept for grid which includes web services driven by intelligent semantic web and semantic grid, there is the need for some kind of environment that will allow the exclusive use of these resources and services by distributed organisations in a secured and coordinated manner.

A problem solving environment (PSE) provides the user with a complete and integrated environment for problem composition and decomposition, solution and analysis for various application areas (Eres *et al.*, 2005). For example, Geodise provides a grid-enabled environment for engineering optimisation through its

graphical components (drag and drop), scripting language, knowledge driven search algorithms and secured authentication security feature. The difference between this research and Geodise is that this research provides a step by step graphical interface for designers to create mathematical models by providing the criteria, design parameters and constraints and then linking the criteria to the design parameters. Additionally, this research provides algorithms as services for users which are first viewed using WebMDS. Geodise uses Matlab for scripting and visualisation capabilities and Globus for security, information and data management capabilities. Eres *et al.* (2004) used the calculation of lift to drag ratio for the flow over a airfoil and shape optimisation for the vibration isolation characteristics of satellite truss designs within the Geodise/Matlab environment to demonstrate how MODO can be effectively done in a grid computing problem solving environment. PSEs expose resources and services to the user to “plug and use”.

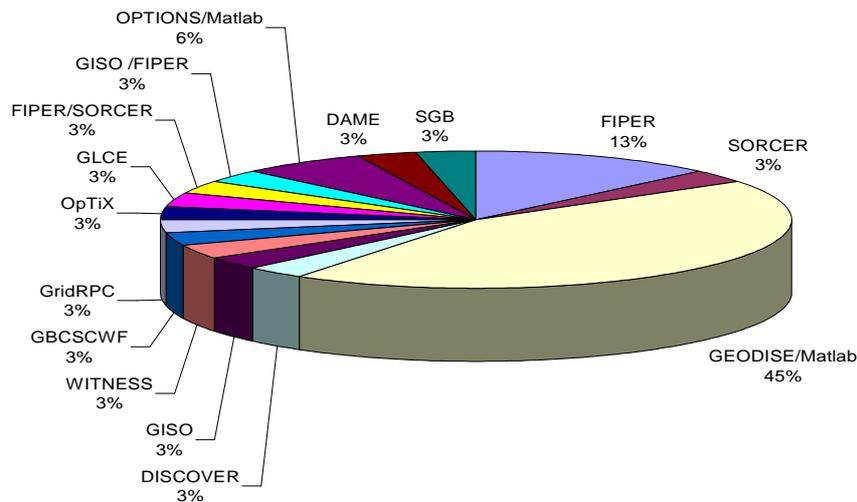


Figure 2.6: PSE pie chart showing PSE tools and Number of Papers/percentage

Figure 2.6 shows the percentage of research papers for different problem solving environments. Geodise (45%) seems to be the most popular in this research and is normally used by researchers in the area of design optimisation. Other PSEs that are also popularly used for design are FIPER (13%), SORCER (3%) and OPTIONS/Matlab (6%). Again this justifies why this research is applying grid services for MODO applications.

2.3.4 Fourth generation of grid computing

The fourth generation of grid computing consists of research activities in semantic grid and its associated workflow management.

2.3.4.1 Semantic grid

Semantic grid incorporates ontologies, knowledge, agent-based applications and inference rules in managing heterogeneous and dynamic resources with high level of coordination of grid services. The semantic grid seeks to incorporate the semantic web into its implementation (Cannataro and Talia, 2004). High level services such as workflow generation and systematic process flow as witnessed in MODO requires the use of intelligent agent-based optimisation environment to aid in search exploration (Gil *et al.*, 2004).

What will complement the present efforts in open grid services framework are the syntactical descriptions of service and application relationships (Li and Min, 2005). The interfaces of interacting services should be able to discover, reconfigure and adapt autonomously to service provision through the implementation of semantic grid (Hau *et al.*, 2003). Protocols such as OWL (Web Ontology Language), WSDL (Web Services Description language), UDDI (Universal Description, Discovery and Integration) and XML (eXtensible Markup Language) are essential for identifying services based on semantic expressed capabilities. Organising the large amount of data produced and analysing it is a challenging task. The emergence of semantic data grid (SDG) is seen as the solution to this problem (Schwidder *et al.*, 2005). The effective automated discovery, analysis and integration of data and metadata in grid environments can only be achieved through some kind of semantic annotation of process flow in MODO (Fox, 2003). The future success of grid computing depends on the availability of semantic/knowledge-rich resources for science and business (Chen *et al.*, 2004). The Semantic grid is the Internet-centred interconnection environment that can effectively organise, share, cluster, fuse, and manage globally distributed versatile resources based on the interconnection semantics coordinating the activities of optimisation engineers within MODO environment (Zhuge, 2004).

2.3.4.2 Workflow management grids

Since the concept of grid is based on sharing computational resources and collaboration to enhance efficiency and cost-effectiveness in science, engineering and

business, the flow of resources across different interdisciplinary domains is bound to occur. To ensure smooth cross-domain collaboration, workflow management services are incorporated into the grid. Workflow management consists of workflow construction, simulation, scheduling, monitoring and conflict resolution to provide services that meet quality of service (QoS) agreements among grid users (Cao *et al.*, 2003).

2.3.5 Future generations of grid computing

There is a lot to be seen in the future of grid computing. There are many unanswered research questions. Such questions include-can grid resources manage themselves in the event that too many resources are part of the grid? Can the grid provide redundancy for all resources on all nodes for use in case of any failure? Besides, grid computing has not yet started yielding the expected benefits for end users. The future research issues will concentrate on extending ideas and implementation of autonomic computing, economic models, ubiquitous computing and cloud computing.

2.3.5.1 Autonomic grid computing

Future grid computing resources and services should be self-adjustable, self-adaptable, self-manageable and self-configurable. Unlike conventional information systems that have few interactions, the grid will have multitude of users interacting and sharing heterogeneous resources dynamically. This presents a great deal of management issues which calls for the resources and services to automatically manage themselves without the intervention of system administrator (Folino and Spezzano, 2007). For example, complex engineering design optimisation in the future will be done within a problem solving environment that has resources located at different centres of expertise and optimisation code to be used will be selected based on the constraints, inputs and parameters of the problem presented without the intervention of the design engineer (Parmee *et al.*, 2005). The interfaces of services and resources configure and adjust themselves automatically. In this way, the efficiency of design experts will improve tremendously. Another example is in the optimisation of oil reservoir placement. The location of wells in oil and environmental applications significantly affect the productivity and economic benefits of surface reservoir (Parashar *et al.*, 2005b). To get the best benefit, the determination of optimal location is important. This calls for autonomic grid computing services that will use

intelligent reservoir simulators and select the best optimisation search for the location of an economic oil location. This grand challenge of systems managing themselves requires specific and generic autonomic computing elements.

2.3.5.2 Ubiquitous grid computing

Ubicomp (ubiquitous computing) is a grand challenge vision which allows people and environments enabled with computational resources to provide information and services when and where desired (Hightower and Borriello, 2001). Grid-enabled ubiquitous computing will lead to systems knowing the physical location of things so that they can record them and report them to those that need to know. For example, the system could help the search and rescue team locate exactly the position of victims and their condition for quick evacuation. This system will revolutionise the field of mobile computing for efficient identification of agents (Forte *et al.*, 2007). This grand challenge computing discipline will enable computers to be aware of our environment and take critical actions which entails resolving middleware challenges such as balancing between transparency and context-awareness and the requirements for certain degree of autonomy (Soldatos *et al.*, 2005). The pervasive nature of grid computing coupled with the multidisciplinary and distributed nature of engineering design optimisation require the use of ubiquitous grids in the future. The strategy will incorporate environmental model specification, content acquisition, content tagging and intelligent network device adaptation (Grady *et al.*, 2007).

2.3.5.3 Service economic model for grid computing

The grid needs to address issues concerning service on-demand and charge per usage. A dynamic metering and accounting system for dynamic usage of resources and services will be more appropriate for grid computing service-oriented architecture. Presently, Web services use static charging system for e-commerce. For example, buying a book from amazon.com is charged per price of the book. The accounting pricing system in grid will be based on many criteria such as the quality of service, policies, time of amount of usage and so on. The dynamic changes in services and resources will also affect the pricing system. Grid Resource Brokers (GRB) component of the middleware can be improved to handle dynamic grid accounting process (Barmouta and Buyya, 2003). MODO is dynamic in nature and as such it requires dynamic coordination of resources in an economic manner.

Computational grid has enhanced the capability of distributed computing systems to model and simulate complex systems in the scientific, engineering and commercial domains (Darema, 2005). The future trend in MODO using grid-enabled technologies will be based on ubiquitous (pervasive) and on-demand service-oriented access to optimisation resources. The notion of dynamic data driven application systems (DDDAS) would see the grid encapsulating services such as the quality of data (QoD) or quality of service (QoS) assessment, uncertainty model services, and sensor services in MODO process. The electricity grid, from which the computational grid was conceived, had an equally visionary focus, with more precise focus on the needs of its immediate market than the computational grid of today. The argument here is that the electric power grid had a well defined set of users that the expected net income would produce a viable investment. Thus providing incentives for the intellectual property associated with its invention and provision as a utility. For the computing grid to assume a global infrastructural status like the electric power grid, it must have a clearly defined service oriented nature with economically active users who can confidently pinpoint and identify associated value or “utility” to that service which should ordinarily exceed value/cost ratio of other existing alternative technologies such as the internet services and should also cover cost of migration from those existing technologies.

In the days of Edison (considered as one of the fathers of electric power grid), there was an existing utility (gas) and it was a matter of getting around the competitive economic advantage over the gas lighting as a benchmark and substitute it with electric lighting. Edison also paid close attention to the needs of various customers and had various sections of his company deal with different products relating to specific customers. For example, Edison Lamp Works produced lamps and Edison Illuminating Company supplied lighting. This organised level of division of service provided a better focus and specialisation that was used as a productivity advantage over other existing competing utilities such as gas lamp. Today’s computational grid is also drifting in the same direction. The initial conception of the grid was mostly for scientific computation and data intensive applications. The present shift in emphasis to a more service oriented grid informed the decision to have different types of grids. There are now computational grids, data grids, enterprise grids, extraprise grids, global grids and many more. Just like in Edison’s companies which serve different

customers and different purposes, these different grids serve different users and different purposes. For example, the computational grid serves users who need computing power for their jobs, data grid serves users who analyse data and use data intensive applications, extraprise grids are for companies and businesses to collaborate with their customers and partners, and global grids will be a service utility for general purpose business such as in e-commerce. The only way to keep the tempo of enthusiasm in grid computing infrastructure is to ensure that its business value, reliability and application performance outweighs today's competing technologies such as parallel distributed technology and web services technology. These values and applications must be visible to the "real" end-users of grid technologies. For example, services should be categorised based on private, commercial or industrial usage. The charge rates should also be based on the same categorisation. Furthermore, services for MODO should be based on simple, medium or complex optimisation problems. This categorisation will serve as an incentive to many users.

2.3.5.4 Cloud computing

The concept of cloud computing is based on offering computational resources (software, hardware, data, information) as service on-demand to remote users. This concept is championed by information technology (IT) giants Google and IBM. Google's MapReduce project (Dyer *et al.*, 2008) and IBM's Blue Cloud project (Hayes, 2008) are both aimed at providing cost-effective computational resources for their use as well as 'outsourcing' part of it to customers that need them and charge them per usage. Cloud computing has proved to be the first truly production 'utility grid' based on service-oriented architecture. To demonstrate the success of cloud computing, retail giant, Amazon is now commercialising its computational resources using the cloud paradigm after testing its Elastic Cloud Computing (EC2) product in 2007 (Weiss, 2007).

2.3.5.5 Nexus

Nexus is a middleware which allows implementation of different applications written in different computer languages to communicate and exchange data in a distributed environment. In this way persistent (long-lived) objects and applications can be dynamically created and managed in a distributed environment (Tripathi *et al.*, 1992). Remote service request by multiple users is handled uniquely by an identifier called a handler. This allows commodity objects to be used in heterogeneous environment that

runs different applications and operating systems (Foster et al., 1995). The Nexus concept is an innovative way of providing applications as services through remote service request operations in grid computing. This allows service on-demand for utility grid.

2.4 Grid projects

It is important to look at some grid projects in this research. This will serve as a motivation for the research. The late 1980s and early part of 1990s witnessed early phase of grid projects initiatives. Appreciable successes were recorded with challenges that sharpened and refocused the original vision of the grid. The Open Grid Forum (OGF) provides an avenue where grid researchers meet to discuss issues on projects. Governments, research institutions and commercial sectors have spent huge amount of resources on various grid projects. Some notable grid projects are listed in Table 2.2 according to the generation of grid computing and its evolution.

Table 2.2: Some grid projects based on generational classification (sources: Hey and Trefethen, 2002; Buyya, 2002; Berman et al., 2003)

Generation	Project	Funded by	Functionality
1st Generation (1989-1993)	I-WAY Globus Legion Cactus GriPhyN PPDataGrid TeraGrid GridCentre Damien GrADS	NSF NSF/Argonna Lab NSF NSF NSF DOE NSF NSF EC NSF	Middleware Middleware Middleware Middleware Computation & data Computation & data Computation & data Computation & data Computation & data Computation & data
2nd Generation (1994-1998)	gLite FusionGrid Earth Systems Grid iVDGL DISCOM Science Grid NEESGrid Condor AppLeS	EC DOE DOE NSF DOE DOE NSF NSF NSF	Middleware Computation & data Computation & data Computation & data Computation & data Computation & data Computation & data Scheduling Scheduling
3rd Generation (1999-2003)	DataGrid EuroGrid DAMIEN GridLab FIPER/ SOCER OGSA GRIP GridLab CrossGrid Grid-Ireland Grid for Remote Computing UNICORE NAREGI DAME GENIUS White Rose Grid Grid-oriented Storage	EC CERN, PPARC, NISP EC EC CRIHAN, CEPBA, EADS NSF NSF NSF EC EC NCCR EC EC Japanese Government EPSRC EC EPSRC/Rolls Royce EPSRC	Computation & data Computation & data Computation & data Computation & data Computation & data Computation & data Middleware Resource Broker Resource Broker Computation & data Collaboration Computational PSE PSE Virtual organisation/PSE Dynamic data storage

4th Generation (2004-2006)	SAM-Grid MyGrid MOBY-Services Semantic-MOBY Gridbus ChinaGrid GRIA Geodise NetSolve DISCOVER SETI@HOME project	PPDG NSF NSF EC Australia Research Council Ministry of education China EC EPSRC NSF EC NSF	Data and computational Autonomic grid Autonomic grid Semantic grid Middleware Resource management Middleware PSE PSE Middleware Virtual reality network
Future Generation (2007-date)	MapReduce Blue Cloud Elastic Cloud Computing (EC2) eDiamond Commodity Grid (CoG)	Google IBM Amazon EC EC	Utility grid Utility grid Utility grid Data consistency Component grid

(CEPBA= Center for Parallelism of Barcelona; CRIHAN= Centre de Ressources Informatiques de Haute-Normandie; CERN= Conseil Européen pour la Recherche Nucléaire; DOE=Department of Energy ; EADS=European Aeronautic Defence and Space Company ; EC=European Council, EPSRC=Engineering and Physical Science Research Council, NCRR=National Centre for Research Resources, NISP= National Institute for Subatomic physics, NSF=National Science Foundation ; PPARC=Particle Physics and astronomy Research Council ; SETI= Search for Extra Terrestrial Intelligence)

2.5 Multi-objective design optimisation

Multi-objective design optimisation (MODO) is the process of improving designs in the presence of constraints and multiple objective functions using classical methods, evolutionary computing (EC) techniques or swarm algorithms (Ray and Liew, 2002). MODO involves the conceptual, preliminary and detailed stages (Roy, 1997). The conceptual stage is when designers conceive the idea and framework for a product. This involves innovation and creativity. The preliminary stage involves analysis of tangible features of the product among multiple options. The designers make decisions to narrow the number of possible options. This is known as divergence and convergence process. The detailed stage involves collective decision making to come up with a final design. The dynamic change in product requirements due to market forces has made researchers to also include dynamic requirement functions for MODO (Roy and Mehnen, 2008). This section will discuss MODO and its methods. The discussion will guide the choice of a MODO method to be used in this research.

2.5.1 Engineering design optimisation philosophy

Optimisation is the art of selecting superior design from a set of feasible solutions based on some pre-defined criteria (Tiwari, 2001). Thus engineering design optimisation deals with the process of obtaining superior designs that satisfy real world requirements and constraints from other possible designs. Examples of real-world problems that use engineering design optimisation are the design of bridges to enhance durability and reliability, the design of aerospace of aerospace structures for better performance and the design of the design of turbines for maximum efficiency

(Tiwari, 2001). To enhance efficiency in the design optimisation process, designers need tools to analyse the variables and run simulations that require computational and collaborative facilities. The choice of algorithm that suits the problem is also essential for better results. The layers of design decomposition aligned with the design processes help in concurrent decision making when done within a Grid virtual organisation (Sobolewski, 2007). This research intends to provide a platform for design experts to build mathematical models and link them to algorithms for optimisation. Problem solving environments are enablers to concurrent engineering design processes, using efficient workflow management systems (Yu and Buyya, 2006). Engineering designs have coupled components that present constraints to design optimisation and so by providing enabling environment for designers to perform optimisation in distributed environment can enhance productivity.

2.5.2 Challenges in MODO

There are challenges that are inherent in MODO. This is perhaps the reason behind devising different methods to solve MODO problems as is explained in the next section. One of the challenges of MODO problems is mathematical model complexity (Papalambros, 2002). This complexity may be due to the non-linear nature of a model. The complexity increases if the model consists of a mixture of discrete and continuous variables. Complex implicit models naturally require much iterative computational process. In this research, the turbine blade cooling system is an implicit model with one discrete variable and eleven continuous variables. Mathematical model complexity is classified into local and global optimisation and surrogate models. Local and global challenge arises if not all the functions in the model are continuous and differentiable with respect to all the design variables (Papalambros, 2002). This situation may give rise to multiple minima. Surrogate models are complex models whose inputs are results obtained from other sub-models. This research proposed the mathematical model building service to help designers overcome such challenges collectively through collaboration. Other challenges of MODO are multidisciplinary involvement of experts, computational and data intensity and choice of appropriate algorithm. To come up with good designs, multidisciplinary experts collaborate and bring in their different creativity and innovation. Experts may be distributed and using different software and hardware systems. Interoperability problem may arise. The middleware concept in grid system is proposed to address this interoperability.

The next section describes and compares MODO methods to come up with an appropriate method that the grid can be used to address the challenges mentioned.

2.5.3 MODO methods and approaches

As has been stated earlier, MODO problems bring together different professionals with diverse backgrounds to collaborate. These professionals are usually distributed in different geographical locations and often need to share resources such as data, information, algorithms, software, hardware and expensive instruments (Foster *et al.*, 2002b). Apart from bringing diverse experts together, sub-problems that have been addressed by individual experts need to be coupled together within MODO framework. Some of these sub-problems depend on the output of one or more sub-problems as their input variables. This means that the start and finish time of sub-problems vary and depend on one another (Suh, 1995). Complex optimisation problems can have very large number of sub-problems with different constraints and objective functions that require multiple numbers of iterations during computation. These characteristics of MODO problems call for different methods and approaches. Evolutionary computing (EC) techniques have evolved as good candidates for solving optimisation problems that require multiple optimum results in a single simulation run (Deb, 2001). This feature enables EC techniques to be used as general-purposed evolutionary algorithms in business, engineering and control systems (Fleming and Purshouse, 2002). This research will introduce the concept of MODO service and recommend a definition for it. This is because MODO experts could be geographically distributed and may be using different software and hardware platforms. This calls for MODO service provision to coordinate the simulation of the different sub-problems. MODO services may include the provision of algorithms, software, hardware, middleware and instruments in a seamless and transparent manner.

Based on the above description a MODO service can be defined as:

a grid-enabled service that consists of tightly coupled sub-services that perform different interdependent functions for different interdependent disciplines engaged in collaborative optimisation process.

Some of these sub-services perform computations on multiple objective functions in one domain subject to other sub-services that compute some constraints in yet different domains. Some of the sub-services produce outputs that serve as inputs to other services. To illustrate this definition in practical grid terms, take for example OGSA and WSRF that ensure the integration of web services and grid services which perform messaging and lifecycle management functions for the web and grid services. OGSA Platform Interfaces (OGSA-PI) must be Open Grid Service Infrastructure (OGSI) compliant and performs functions not defined in OGSI such as defining high-level interfaces of common grid services (Berman *et al.*, 2003). Both OGSI and OGSA-PI (sub-services) are encapsulated in what is called OGSA Platform (OGSA-P) as the main service. This research will use common interface conventions to operate services depending on one another.

In this section, EC and classical optimisation methods are explored with regard to their suitability for MODO applications.

2.5.3.1 Evolutionary computing

Evolutionary computing (EC) consists of techniques such as genetic algorithm (GA), genetic programming (GP) and evolution strategy (ES) which mimic nature (based on Darwin's theory of evolution) to improve the search process of multi-objective optimisation problems (Deb, 2001). The ability of EC techniques to encapsulate more than one objective functions and produce multiple optimum solutions at a time make them attractive in real world optimisation applications. MODO is driven by the need to meet multiple objective needs of diverse customers in today's competitive global economy (Wang *et al.*, 2005). These objectives include improvement of quality of products and services, minimisation of cost, maximisation of profit, reduction of time to market, improvement in customer relationship and many more. These objectives are addressed by different distributed experts ranging from design engineers, economists, marketing and sales men, accountants, statisticians, cost engineers, software developers and others. Constraints and boundary conditions in MODO can be physical (e.g. geographical distribution) or interdependent sub-programs. EC techniques are suitable for MODO problems because of the multi-objective and dynamic nature of MODO applications. This also makes the decision making more objective-driven using the multiple trade-offs that are considered by different experts.

This research has considered using EC techniques for these reasons and will discuss them in more details later. However, non-EC techniques will be mentioned in the next subsection and it is discussed why they will not be used within the context of MODO.

2.5.3.2 Non-EC techniques (Classical methods)

Early optimisation methods are called classical or non-EC methods. These methods produce single optimal solution in a single simulation using both single and multi-objective optimisation processes based on preference-based methods to reduce multi-objective functions to a single objective function. Such methods include weighted sum, ϵ -constraint, weighted metric, Benson's method, value function, goal programming, and interactive methods among others (Deb, 2001). These methods usually involve manual interference in choosing some parameters such as the weight of an objective or constraint based on its perceived importance. For this reason, classical methods are best suited for optimisation of single-objective functions. One advantage of classical optimisation methods is that they are simple to use but share the disadvantage that they are not suitable for multi-objective optimisation problems (Deb, 2001). For this reason, this research intends to use EC techniques rather than classical methods for multi-objective optimisation. This will be discussed in the next section when EC and non-EC techniques are compared.

There are also algorithms known as the Tabu search and heuristic methods. Tabu search is a method for solving combinatorial MODO problems using adaptive approach to combine many methods (Glover, 1989). It is used in solving problems such as scheduling and cluster planning using linear programming approach. Heuristic optimisation method is used to find good feasible solution in circumstances where it is not possible to get exact solution in complex optimisation problems (Rardin and Uzsoy, 2001).

2.5.3.3 Comparison between EC and Non-EC techniques

As mentioned before, both classical and EC techniques have their advantages and disadvantages. Classical algorithms usually adopt a deterministic approach in finding optimum solution (Deb, 2001). This is a good feature if optimisation experts have prior information on what they want. In this case a point-by-point search that is directed towards the region of interest is pursued. Most classical methods are simple to perform and perhaps that is why they seem to be popular among some users.

However, classical algorithms are meant to solve particular types of problems. This means that they can be good at solving one problem and bad at another. Besides, classical algorithms do not have the robust capability to solve optimisation problems with discrete search space and cannot take advantage of parallel high performance computers since they can only produce a single solution in a run (Deb, 2001). Most real world problems have multiple and conflicting objectives and some variables are discrete in nature. Evolutionary algorithms tend to work on population of solutions to produce a set of Pareto-optimal solutions (Zitzler and Thiele, 1998). EC techniques can handle both linear and non-linear problems with discrete, continuous or mixed search spaces (Michalewicz *et al.*, 1996). Classical methods hardly handle efficiently optimisation problems that are mixed with minimisation and maximisation objectives. EC techniques have inherent parallelism and produce multiple solutions in a single run and so are suitable for use in any parallel high performance computer (Zitzler, 1999).

2.5.4 Adoption of evolutionary computing approach

As discussed above, the deterministic nature of classical optimisation methods using either direct or gradient-based techniques makes them inadequate for multi-objective optimisation problems. This is because most classical algorithms depend on the initial trial solution, do not get to completion in some cases, are not generic and sometimes cannot be run efficiently on parallel distributed machines (Deb, 2001). Evolutionary algorithms can take advantage of parallel processing power of the grid to produce multiple optimum solutions in a single simulation run. Grid computing is a large-scale distributed computing paradigm which can handle multi-objective functions that are computed from different geographical locations on heterogeneous platforms ensuring efficient compute resource utilisation, sufficient optimisation algorithms and integration of the algorithms within a virtual organisation (Grauer *et al.*, 2004). Based on this and the drawbacks of classical algorithms as stated earlier, this research focuses on evolutionary algorithms for MODO. However, the grid platform accommodates any algorithm in the form of a service. The grid service developed in this project handles this requirement. This research will use an EC technique for the implementation of the 3 case studies. A brief description of the main EC techniques is necessary to justify the reason for using them here.

2.5.4.1 Genetic algorithms

Genetic algorithms (GA) mimic nature to perform search operations space using crossover and mutation on binary coded strings (chromosomes) that represent decision variables in an optimisation problem. These operations (crossover or mutation) produce a number of results (off-springs) which represent better solutions based on the constraints and objective functions under consideration. This refinement continues with successive duplication of good solutions and dropping of bad ones using reproduction operation and maintaining the population constant until sufficient convergence is reached to terminate the iteration of the operations (Deb, 2001). This is similar to biological theory of evolution which states that successive offspring have better features than their parents and performed better in competitive survival of the 'fittest'.

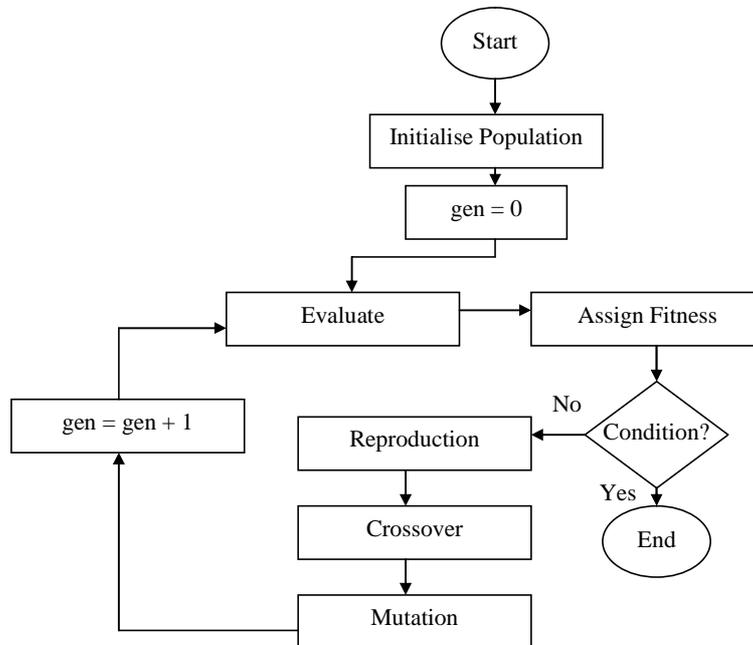


Figure 2.7: GA flowchart (Deb, 2001)

GA produces multiple optimum solutions in a single simulation run. This is the beauty of its applicability in MODO. In MODO services, there is a set of optimisation services published by service providers and each is expected to have some degree of autonomy over its services. Autonomy is a constraint in grid-based service provision and GA can be one of the services provided by a service provider for optimisation. Multi-objective GA can be used in allocating these services on-demand to

multidisciplinary professionals within MODO as well as providing GA codes as services (Parashar and Bloebaum, 2006).

2.5.4.2 Genetic programming

The implementation of GAs into computer programs to solve optimisation problems is known as genetic programming (GP). In MODO, different software programs are used by different professionals producing multiple solutions that are matched against desired criteria. Differential equation methods and statistical methods are applied in GP to mimic the crossover and mutation operations in GA. These operations are carried out on solution of each function. Multi-objective optimisation problems consist of many functions and sub-functions. This means that GPs can consist of multiple programs that are used from different geographical locations and their results merged for comparison. Grid platform provides a scalable computing environment for different sub-programs that relate to different aspects in MODO problem. The solutions (programs) are tested and compared to find a fitness function. Service-oriented grid environment aims at publishing different GP programs as services that can be accessed and used by multiple users engaged in MODO tasks.

2.5.4.3 Evolution strategies

Evolution strategy (ES) is similar to GA but uses more mutation related operations on real values. Because ES uses a range of real values based on recombination of the values, it is computationally expensive. Grid-based computation on multi-objective optimisation uses idle computing cycles and provides on-demand computing power as services. MODO service provision uses Condor software to pull together desktop compute cycles to overcome computationally intensive jobs.

2.5.4.4 Evolutionary programming

This is basically a mutation oriented algorithm. Evolutionary programming (EP) is applied on real parameters similar to some aspects of ES. Self adaptation is used during mutation to reflect a change in mutation capability. Dynamic MODO service provides adaptive features and distributed probability programs as services.

2.5.4.5 Other evolutionary methods

The disadvantages of using one type of algorithm to solve different problems made researchers to come up with other optimisation algorithms. This research will only mention some of them as they are beyond the scope of this study. These algorithms

tend to improve upon the existing algorithms for solving particular problems as they have a combination of two or more features from other algorithms. Examples of such optimisation algorithms are vector evaluated genetic algorithm (VEGA), niched-pareto genetic algorithm (NPGA), random weighted genetic algorithm (RWGA), multiple objective genetic algorithm (MOGA), predator-prey evolutionary strategy (PPES), distributed sharing genetic algorithm (DSGA), hybrid particle swarm optimisation (HPSO) (He and Wang, 2007).

2.5.5 MODO service approach

Based on the MODO methods discussed above, this research intends to adopt a service-oriented approach for providing computational resources to enable distributed MODO experts take advantage of grid-based optimisation service environment. The Grid Information Service (GIS) and Monitoring and Discovery Service (MDS) support diverse resource groupings, discovery and search procedures at the same time allowing users to control inputs and outputs (Czajkowski *et al.*, 2001). This diversity of resource management allows the diversity in objectives and variables used in optimisation algorithms to be managed within the grid environment. Resource sharing takes into account the different optimisation tools and needs of multidisciplinary fields usually involved in MODO. This means that collaborating researchers in the field of MODO can each contribute computational resources to be used by all in the virtual organisation (Czajkowski *et al.*, 2001). Using this capability of the grid, this research has proposed to provide services for computational resources, parameter input service, optimisation service and collaboration service. These services will be discussed in details in chapters 5, 6, 7 and 8. The services will use non-dominated sorting genetic algorithm-II (NSGA-II) to run multi-objective design optimisation problems as case studies (Deb, 2001). The next section will discuss grid as an enabler for MODO.

2.5.6 Grid as an enabler for MODO

The grid provides an enabling problem solving environment for MODO. For example solving a MODO problem may require the integration of heterogeneous distributed software and hardware to carry out optimisation, visualise the results and exchange data (Xue *et al.*, 2004). For example, the third case study (design of a manufacturing plant layout/floor planning-see section 8.6 in chapter 8) in this research uses a

visualisation package that accepts its inputs from NSGA-II objective (quantitative) computations to produce the plant design pattern while allowing the design expert to rate the quality of the design. These ratings (qualitative objective) are fed back to the NSGA-II and the next sets of designs are produced. This continues until the last generation. This is made possible through the grid robust middleware and information service that enable heterogeneous sharing of optimisation resources. The computational and data capability of the grid also serve as an enabler to MODO problems which are naturally data and computationally intensive. An example of a grid enabling problem solving environment is the Geodise (Grid_Enabled Optimisation Design Search for Engineers) toolkit which provides a grid environment that allows optimisation engineers to deliver a better design within restricted time scale, reuse previous models and knowledge (Song *et al.*, 2004). This research aims to improve on this by including interfaces for mathematical model building and linking the model directly to an optimisation algorithm, thus creating more efficiency and speed of computation. In addition, businesses such as manufacturing, aerospace, and automotive sectors which need to work with heterogeneous partners stand to benefit a lot if successful grid protocols are incorporated into their daily transactions and processes. The Metacomputing Directory Service (MDS) of the Globus Toolkit (GT) can be used as a strategy to integrate agile manufacturing into production facilities to improve the speed of production (Jie and Jian-gang, 2004). Jie and Jian-gang (2004) proposed a concurrent approach of allocating resources to tasks in manufacturing processes for optimum efficiency. This was accomplished by using the rich information provided by the Grid Resource Information Service (GRIS) and Grid Index Information Service (GIIS) of the MDS within a tailored architecture for agile manufacturing. GRIS is the database server that stores all data and information on network resources and services uniquely for each grid node. This means that there is local GRIS on every node in the network. For redundancy and robust reliability, all information on every GRIS is aggregated in GIIS. GIIS is the global information reservoir in grid systems. A user can query GRIS for faster retrieval of information or query GIIS depending on proximity of the two to the user.

Having discussed the grid as an infrastructure to facilitate MODO applications, it is important to study some features of existing problem solving environments, some of which are used for optimisation.

2.6 Grid-enabled problem solving environments

This section describes some related problem solving environments (PSEs) that are used for engineering design optimisation or concurrent engineering. This is to give the research an opportunity to learn important grid features that are suitable for optimisation problems as well as to identify the challenges that PSEs faced to solve optimisation problems. Although there are many PSEs, only 5 will be described within the context of this research. This is because from literature and industry survey (see Figures 2.3, 2.4, 2.6, 4.5 and 4.6 in chapters 2 and 4 respectively), these PSEs are the most favoured by researchers in design optimisation. For example, Globus is the defector middleware for most grid researchers.

2.6.1 Geodise

Geodise is one of the numerous UK e-Science projects which aim to provide grid-based seamless access to intelligent knowledge repositories, state-of-the-art collection of design optimisation and search tools, analysis codes, distributed computing services and data resources for multi-objective design optimisation (<http://www.geodise.org/>). The project was funded by the UK EPSRC (Engineering and Physical Sciences Research Council) for an initial period of 3 years from 2001 to 2004 with a grant value of £2,872,450 (\$5,084,237). The project is based at the University of Southampton in UK with collaborating research partners at Universities of Oxford and Manchester also in UK. Collaborating companies are BAE Systems, Rolls Royce and Microsoft UK Ltd. The basic components of the Geodise Toolkit are application services, collaboration toolkits, data mining and analysis services, data management services, domain ontology and metadata, workflow services and problem solving environments. Geodise is usually used in conjunction with Matlab for its scripting and visualisation features. Matlab also have optimisation algorithms such as GA (Genetic Algorithm) for optimisation and search applications. Geodise has proved to be suitable for engineering processes involving computational fluid dynamics (CFD). The use of Geodise tools by the engineers is facilitated by intelligent design advisors targeted initially at CFD (Pound *et al.*, 2003). Geodise is open source

software and is freely downloadable at the Geodise website. Figure 2.8 presents a simple Geodise architecture.

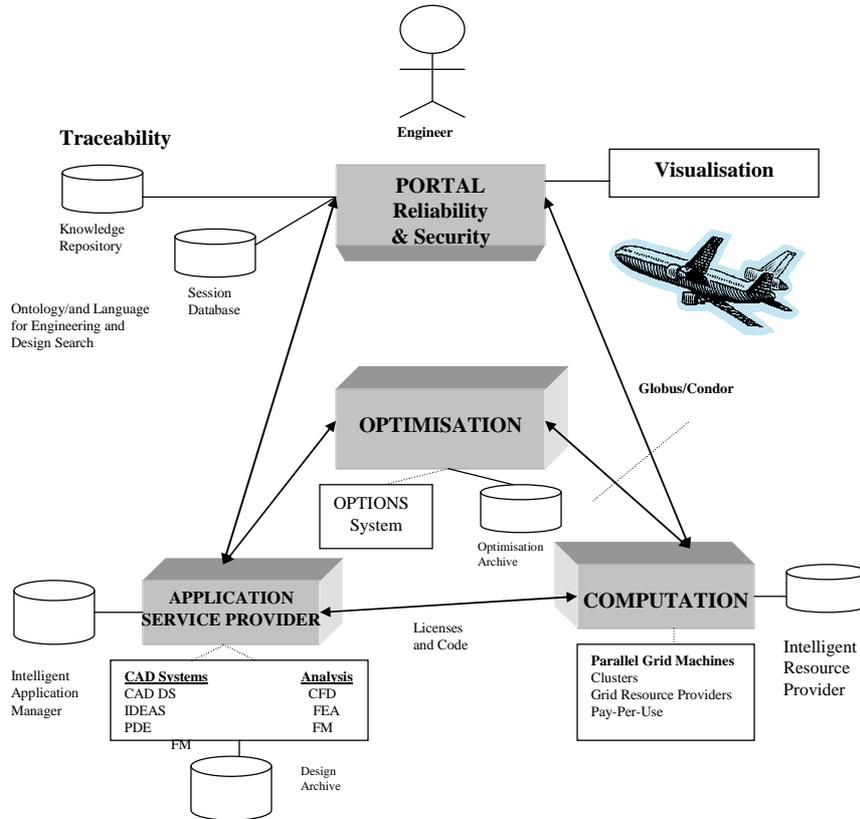


Figure 2.8: Simple Geodise Architecture (Cox *et al.*, 2001)

Geodise project is an attempt to improve the efficiency of design engineers by providing them with optimisation tools as grid services (Cox *et al.*, 2001). The portal allows authorised design engineers seamless access to the system and the databases allow input and output operations managed by a service. The designer is guided through an optimisation knowledge-based service. The system allows incorporation of commercial systems such as CAD (computer aided design) and analysis tools by application providers as services. This demonstrates how design optimisation can be made more efficient by making search and optimisation tools handy to designers (Eres *et al.*, 2004).

2.6.2 FIPER

FIPER (Federated Intelligent Product Environment) is a 4-year project sponsored by the National Institute for Standards and Technology-Advanced Technology Program (NIST-ATP) in the U.S. Its aim is to produce an intelligent system that leverages the

emerging web technologies in which engineering tools such as CAD (Computer Aided Design), CAE (Computer Aided Engineering), PDM (Product Data Management) and optimisation algorithms (Rohl *et al.*, 2000) act as distributed service providers as well as service requestors communicating through intelligent context models for concurrent engineering design optimisation. FIPER has a grant worth of \$21.5 million with Ohio and Stanford Universities as academic partners and GE (General Electric) teaming with Engineous Software, BFGoodrich, Parker Hannifin and Ohio Aerospace Institute as company partners. FIPER uses GE Aircraft Engine functionalities to demonstrate the capturing of designer's knowledge in Knowledge Based Engineering (KBE) systems to create Intelligent Master Model (IMM). This IMM contains the 'what', the 'why' and the 'how' of a design (Rohl *et al.*, 2000) using multi-objective optimisation algorithms to produce a range of global optimal solutions. FIPER provides a graphical environment that permits interactive click-and-drag Grid programming interface which users can reuse and execute concurrently (Sobolewski and Kolonay, 2006).

2.6.3 SORCER

SORCER (Service Oriented Computing Environment) is an extension of the work of FIPER project. SORCER lab is based at the Texas Tech University, U.S.A. The goal of SORCER is to form Grids of distributed services that provide engineering data, applications, tools and evolutionary computing (EC) optimisation algorithms (Soorianarayanan and Sobolewski, 2004) for concurrent engineering design disciplines.

2.6.4 DAME

DAME (Distributed Aircraft Maintenance Environment) is also a UK project which provides an intelligent grid demonstrator system for health monitoring and fault trouble-shooting in mobile aircrafts. The need for mobile devices to have secured and seamless access to Grid services while actively working on the field is important in both scientific and business disciplines (Ong *et al.*, 2005). In this framework, optimisation algorithms are used to select optimal diagnostic solutions based on certain constraints and parameters using multiple objective functions for the optimisation. The DAME project develops grid-based fault diagnosis and prognosis for aircraft maintenance (Jackson *et al.*, 2003). This is done by capturing data and

information on various parts of the aircraft and providing this information to maintenance engineers as grid services when there is a deviation from normal behaviour. This helps engineers to take prompt decision.

2.6.5 Globus toolkit

Globus is an open source toolkit considered as the de facto middleware for large-scale computing services. The Globus toolkit is funded by various organisations such as the Defense Advanced Research Projects Agency (DARPA) USA, National Science Foundation (NSF) USA, Globus Alliance and the UK e-Science program. The I-WAY (Information Wide Area Year) project which succeeded in linking many supercomputing centres was the first initiative that led to the development of the Globus toolkit in 1995. Globus toolkit consists of many services and protocols such as Globus Security Infrastructure (GSI), Globus Information Services (GIS) and Globus Resource Allocation and Management (GRAM). Globus has evolved to the current version 4.0 with many additions to the protocols DAI (Data Access and Integration) developed by the UK e-Science Centre and the standard ‘plumbing’ features of the OGSA (Open Grid Services Architecture) which provides standard implementation interface for participants (service providers and service requestors) in the Grid so as to ease and facilitate interoperability among heterogeneous resources and services (Foster and Kesselman, 1999). Globus works on a layered architecture. The lowest layer is called the fabric layer. This layer manages the core hardware and operating systems as well as shared computational resources (Foster *et al.*, 2003). This layer has the capability to implement local specific MODO resource sharing using shared files and data catalogues. The next layer is the connectivity layer which is responsible for the network and connectivity authentication protocols such as TCP/IP (Transmission Control Protocol/Internet Protocol) for secure communication. It also uses APIs (Application Programmer’s Interfaces) and SDKs (Software Development Kits) for programmers to write communication codes. The third layer is the resource layer. This layer consists of core grid shared resources and it does initiation, monitoring and control of sharing individual resources within the grid community. This layer is where optimisation resources contributed by individual users are managed separately. The fourth layer is the collective layer which coordinates multiple sharing of resources. This layer is crucial as the whole resources in the grid are managed here. The fifth layer is the application layer. This layer is the one that interfaces directly with the end

users and it consists of applications. This is the layer where the design parameter input service is run to present input interface to designers. To join in the Grid, local users must follow the standard protocols to reduce customisation efforts (Foster *et al.*, 2001). This philosophy encourages autonomic grid resource management. This research is using Globus as its middleware for implementing services for multi-objective optimisation. Figure 2.9 shows the schematic diagram of Globus architecture.

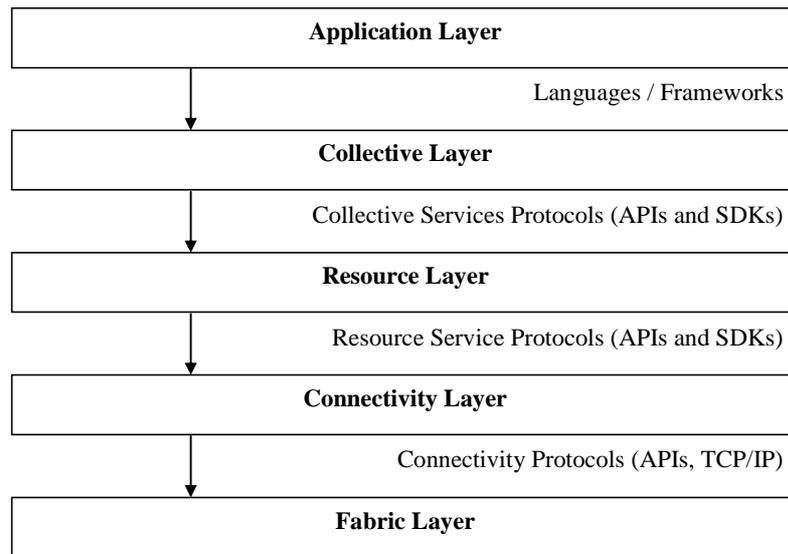


Figure 2.9: Globus Layered Architecture (Foster and Kesselman, 1999)

2.6.6 Challenges that PSEs face in solving MODO problems

Now that the research has studied some PSEs, it is essential to look at problems that PSEs face in solving MODO problems. One of the major issues in the implementation of PSEs for MODO is dynamic and coordinated resource sharing in the presence of heterogeneous resources under high level of dependability. Although such capabilities are part of the OGSA, it is difficult to attain an appreciable level of coordination among distributed service composition and selection (Cheung *et al.*, 2004). For example, if a resource on node A happens to be unavailable when node B needs it, what happens? This is an active area of research to ensure QoS to grid users. However, researchers are looking at providing redundancy in the form of grid overlay architecture that could take care of grid service failures (Grace *et al.*, 2004). Computational and data adequacy for MODO applications is also still not met, even with the cycle-stealing concept using Condor scheduler. For example particle physics and multi-objective optimisation of complex systems using grid systems still present

challenges due to constraints such as global and local policies, dynamic resource utilisation and inadequate response time (Ranganatan and Foster, 2002). In this research, the concept of single population parallelisation is adopted to ensure efficient and speedy optimisation process. Single population parallelisation is the process of sending sub-parts of the optimisation process to different nodes with all sub-tasks sharing the same population and common parameters. Though Globus and other middleware provide good authentication and authorisation capabilities, different companies and researchers prefer different models to be adopted when building a PSE. This is why there is hardly any PSE that serve a generic purpose; most PSEs are developed to solve particular or specific problems. PSEs that are built for commercial purpose are restricted to legacy middleware because of the security issues. The GRIA project is one of such examples. In this research, the Virtual Organisation Membership Service (VOMS) is configured in Globus to assign different authorisation and authentication rights to different users based on the needs of each user. This concept allows users to have control over their resources. One aspect that is often ignored is the support and training aspect during PSE implementation. In this research, the proposed service specification document describes the negotiating terms between providers and users. These terms which are under the service level agreement includes training, support, availability, licence renewal and service updates.

Having identified some problems facing PSEs, the next section intends to identify the process and procedure for defining a grid service within PSEs such that certain precautionary measures are taken to address some of the problems.

2.7 Grid service definition

The business wisdom in packaging products or services in a manner that makes them visible and appealing to prospective buyers or clients is increasingly used in more innovative ways in recent times. Visibility in terms of quality, quantity, cost and usefulness are important parameters to attract clients to a product or service when they are searching from multitude of the same kind of competing products or services offered by different providers (Abbas, 2001). Grid services are no exceptions to these concepts. In this section, the research explores the essential constituent characteristics of grid services (grid service providers and requestors), how a grid service is defined, the process of grid service definition and the representation of grid service definition

to users according to the Open Grid Services Architecture (OGSA) standard and Web Services Resource Framework (WSRF) specifications (Berman *et al.*, 2003). These two standards define the approaches for creating, naming and managing lifetime instances of services. They also take care of declaring and inspecting service state data, asynchronous notification of service state change, representing and managing collections of service instances and common handling of service invocation faults using Grid Services Description Language (GSDL) and Extensible Mark-up Language (XML) (Foster *et al.*, 2002a). XML is a web-based language which allows machine-to-machine communication and enables the usage, sharing and data representation in a structured manner over the internet. As explained before, services are developed and implemented using GSDL, which is an XML-based language. GSDL is an extension of Web Services Description Language (WSDL) to cater for grid service implementation. WSDL is also an XML-based language that performs the same functionality as GSDL. The researcher makes use of the knowledge obtained in studying PSEs and optimisation services to come up with a list of critical variables that make grid services practically useful. These critical variables are discussed in section 6.4 of chapter 6. Some of these variables are also identified through industry survey of some commercial companies.

2.7.1 What is a grid service

Grid services (GS) are extensions of web services (WS). As has been stated before, a web service as defined by the World Wide Web Consortium (W3C) is a software system designed to support interoperable machine-to-machine interaction over networks using interfaces expressed by WSDL or GSDL which allow other systems to interact with them (interfaces) using protocol such as Simple Object Access Protocol (SOAP). SOAP is the protocol specification that enables structured information to be exchanged as services over grid or web-based applications. SOAP uses the application layer in grid architecture to expose its functionality. A Grid service is a stateful and dynamic web service expressed by Grid Services Description Language (GSDL) and run side by side web services using WSRF as platform (Berman *et al.*, 2003). As stated earlier, WSRF is the protocol that enables grid and web services to interact and run on the same platforms. Every resource in the grid environment is a service and every service is owned and published by a grid service provider. Parameters such as quality of service (QoS), cost of service (CoS), reliability and durability are used by

service requestors to subscribe for services. The section below discusses the steps identified for defining services.

2.7.2 How to define a grid service

A grid service is a software agent that performs some well-defined set of operations (service provision) and can be invoked by users (humans or other services (called agents)) through a high-level feature called network-addressable interface via standard grid protocols and data formats (Buyya, 2002). Network-addressable interface is the Internet Protocol (IP) address of a host or node in a network which uniquely identifies the machine and allows other machines to communicate with it. Standard grid protocols include SOAP, WSDL and WSRF as explained earlier. These services or agents coordinate their activities by sending and exchanging messages within a distributed network system known as service-oriented architecture (SOA). An SOA is a distributed system based on open-standard architecture in which the agents are services that perform specific functions and operations. Resource owners (grid service providers) in the grid environment need to publish services for resource users (grid service requestors) to access. Both resource owners and users need to agree on policies and rules guiding the grid ‘market’ place (Buyya *et al.*, 2001). The next sections will describe the process and representation strategies that grid service providers need to follow to define grid services that meet the clients’ requirements.

2.7.2.1 Representation of grid service definition

Grid resources (services) are defined and represented using standard mechanisms for creating, naming and discovering grid service instances. These standard mechanisms are defined by the Open Grid Forum (OGF) based on Open Grid Standard Architecture (OGSA) and Web Services Resource Framework (WSRF) specifications. OGSA is the architecture upon which grid services are deployed. WSRF provides the infrastructure for the implementation of OGSA functionalities. The representation ensures grid service location transparency, technology neutrality, loosely coupled systems and multiple protocol bindings for service instances and supports integration with underlying native platform facilities (Foster *et al.*, 2002b). The representation of service definition is built on the service model called service-oriented architecture (SOA). Figure 1.1 in chapter 1 shows the basic diagram of SOA. It is the relationship between three fundamental participants in grid service marketplace namely grid

service provider, grid service requestor and grid service discovery agent (Papazoglou, 2003). The service provider hosts and defines the service descriptions as well as publishes the described services for service requestors to access or subscribe for published services. Service requestors find services through service discovery agents in the form of registry such as UDDI (Universal Description Discovery and Integration) protocol which is created and maintained by the service provider. It is important to note that service provider, service requestor and service discovery agent in this context are all software agents. The fundamental logical representation of SOA consists of service interfaces (software agents) and strategies that allow implementation of business functions. In this research, services for MODO applications are implemented using OGSA within the Globus middleware. Below is the hierarchy or taxonomy of the representation of service definition in SOA.

Grid service providers

Providers are software agents that provide services while requestors are software agents that use those services. An agent can play the role of service provider as well as service requestor. Providers publish descriptions of services they provide within a standard layered architecture. For example, a resource such as sensor will be published as a core resource at the lowest layer of representation. Core resources are services that interact with hardware and operating systems and their implementation are not visible to the user. A resource such as CAD (Computer Aided Design) software will be published at the topmost (application) layer so that end users can use it directly. This layered architecture follows the conventional grid architecture as described in Figure 2.9. DECGrid developed in this research uses the same convention to run its services within the Globus environment.

Grid service interfaces

The interaction between service providers and service requestors is done through interface implementation. They simply provide the mechanism through which services communicate with applications and other services. The service interface is the descriptive GSDL implementation of operations available to service requestors to invoke services (Smith and Lumb, 2001). Various OGSA platforms implement different service interfaces as described below.

WSRF: Web Services Resource Framework

WSRF represents the convergence of grid services and web services. It is the interface that implements protocols that enable grid and web technologies to interact on the same platform. WSRF provides capabilities to define mechanisms for creating, naming, service lifecycle, managing and exchanging information among grid services. Because grid services are dynamic and stateful, a mechanism is used to globally assign unique names to every service instance called grid service handle (GSH). This means every service instance is unique and can be identified through its GSH by users. GSH carries no protocol or information and so if a network fails with a service instance running, the same GSH can be used to store information of a new service instance when it is restored. The information in every GSH is encapsulated in a mechanism called grid service reference (GSR). Unlike GSH, a GSR can change over its lifetime if the information on GSH changes. This means GSR can expire because it has an explicit lifetime. WSRF implements stateful grid services, inheritance of service interfaces, asynchronous notification of state changes, references to service instances using GSR and the unique global identifier for every grid service using GSH (Slomiski, 2005).

An important aspect of WSRF is to implement new grid service interfaces called portTypes. Examples of portTypes are GridService (encapsulates the root behaviour of service model), HandleResolver (mapping from a GSH to a GSR), ServiceGroup (allows requestors to maintain group of services), ServiceGroupRegistration (allows grid services to be added and removed from a ServiceGroup), ServiceGroupEntry (defines the relationship between a grid service and its membership within a ServiceGroup) and Factory (standard operation for creation of grid service instances). Most portTypes are optional, but for a grid service to be WSRF-compliant, it must implement the GridService portType (Baker *et al.*, 2005). PortTypes use some elements to implement operations on grid interfaces. For example, the operation findServiceData is defined for GridService portType to query the service data element (SDE) referred to as ServiceData. SDE is the property of any data represented in grid and can be used to query instance of that data. It is this same operation that allows optimisation experts to query available optimisation algorithms and computational resources. For example to use the OGSA service browser graphical user interface to

query NSGA-II optimisation algorithm as a resource, the following command is issued:

```
mds-find-service-data –service http://isxp1313c.sims.cranfield.ac.uk:1081/nsga-ii
```

OGSA Platform Interfaces

The OGSA Platform Interface (OGSA-PI) defines important functions of grid services. These functions include Service Groups and Discovery Interfaces (two-level naming based on GSHs and GSRs provides a way of accessing known services which requires a higher level abstraction to allow clients to find services at lower level), Service Domain Interfaces (grid service collections that produce a high order grid service interface), Security, Policy (agreement between service providers and requestors), Data Management Services and Metering and Accounting (metering interface, rating interface, accounting interface, billing and payment interface) among others (Berman *et al.*, 2003).

OGSA Platform Models

OGSA-PI does not provide capabilities for managing virtual and physical entities (hardware and software), though it provides some basic management mechanisms for grid services. OGSA Platform Model (OGSA-PM) provides set of common models for describing and manipulating real entities such as Linux high performance computing (HPC) cluster to be presented as grid services. Each model defines specific attributes of a domain. For example, a model will define Linux HPC cluster which describes the attributes of Linux clusters that are specific to HPC and are needed to run jobs on clusters. This capability is used in the proposed DECGrid together with Condor to provide statistics on the computational usage of the cluster at any given time.

OGSA Platform Profiles

OGSA Platform Profile (OGSA-PP) defines protocol bindings, hosting environments and domain-specific services.

Every service has a number of service data elements (SDEs) from all the portTypes the service inherits. All services must have logical XML document containing these

values together with a root element type of serviceDataValues. The service implementation can store these values in any format but must convert the internal representation to XML as necessary (Berman *et al.*, 2003). The service specification document that is used to develop the proposed DECGrid services explored this capability. A description of portTypes and SDEs are shown in Tables 2.3 and 2.4.

Table 2.3: PortTypes for Basic Services (Berman *et al.*, 2003).

PortType	Description
GridService	Encapsulates the root behaviour of the service model. A web service must implement the GridService interface. All other portTypes are optional.
HandleResolver	Mapping from a GSH to a GSR
NotificationSource	Allows clients to subscribe to notification messages
NotificationSubscription	Defines the relationship between a single NotificationSource and NotificationSink pair
NotificationSink	Defines a single operation for delivering a notification message to the service instance that implements the operation
Factory	Standard operation for creation of Grid Service instances
ServiceGroup	Allows clients to maintain group services
ServiceGroupRegistration	Allows grid services to be added and removed from a ServiceGroup
ServiceGroupEntry	Defines the relationship between a Grid Service and its membership within a ServiceGroup

Table 2.4 describes the interfaces that services use to communicate among themselves.

Table 2.4: Service Data Elements (SDEs) (Berman *et al.*, 2003).

SDEs	Description
Interface	A list of the names of all portTypes implemented by the service
serviceDataName	A list of names of all SDEs supported by this service instance
factoryLocator	A service locator that points to the grid Service instance that created this Grid Service instance
gridServiceHandle	Zero or more GSHs of a Grid Service instance
gridServiceReference	Zero or more GSRs of a Grid Service instance
findServiceDataExtensibility	A set of operation extensibility declarations for the findServiceData operation. The client can use any of the listed inputelement types
setServiceDataExtensibility	Operation extensibility declarations for the setServiceData operation.
terminationTime	The termination time for the service

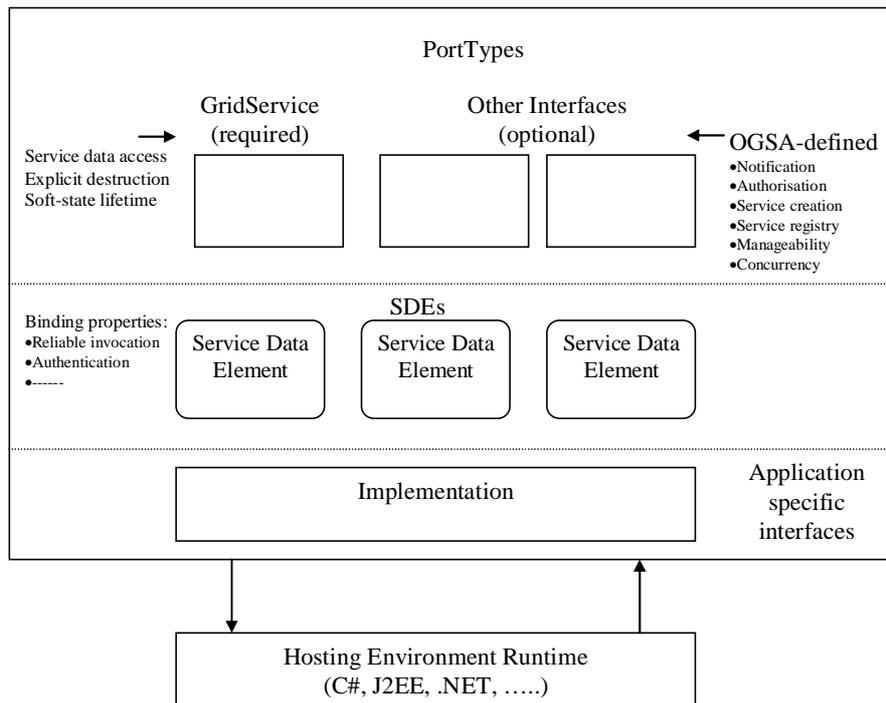


Figure 2.10: OGSA Grid Service (Foster et al., 2002a)

The implementations and configurations of these portTypes, SDEs and operations can be done within the Globus XML schema to suit users' requirements. The development of grid services can be done using two main models. These models are the single-resource grid service model and factory pattern grid service model. Both models use the OGSA standard interfaces. The single-resource model uses the same class to define and implement the service (functionality) and resource (data). This implementation model is only good for small web services. For complex grid services which require multiple data sources and services for different users, the factory pattern model provides different classes for data and services. This makes system maintenance of the grid services easy. This research adopts the factory pattern model because MODO experts need to have different data sources for different services. A simple description of the implementation is shown in Figure 2.10.

MODO Services

MODO framework needs to support a diverse set of services as well as a diverse group of grid users. The diverse services may run on different hardware architectures (HP, Intel, Sun, etc), operating systems (Linux, Sun Solaris, Windows, Mac OS, etc), data formats (relational databases, file systems, etc) and optimisation codes

(evolutionary algorithms, classical algorithms, etc). The diverse group of users may constitute toolkit developers, grid administrators, application users, managers and professionals in the fields of science, engineering and business. Every user has a different need. The domain administrator is concerned with having control over the resources under his hosting environment without being restricted to a centralised administration. In this way he/she can decide who accesses what and when. The application programmer wants to have transparent access to development tools without bothering to have prior knowledge of the hardware or operating system architecture.

When creating new services, a user application issues a *create Grid Service* request on a factory interface, leading to the creation of a new *instance* (Berman *et al.*, 2003). This newly created instance will now be allocated some computing resources automatically. An initial lifetime of the instance can also be specified prior to its creation and allows the OGSA infrastructure to keep the service ‘alive’ for the duration of this instance. This is achieved by sending it *keepalive* messages. The newly created instance is assigned a globally unique identifier called the Grid Service Handle (GSH).

Organisations can come together to collaborate at different granularities. Horizontal MODO service collaboration involves organisations and professionals in similar disciplines, vertical MODO services involve organisations from different disciplines but with complementing roles to produce products or deliver services. Using the above capabilities, a simple implementation of MODO services could involve the following steps:

- Write a portType definition
- Write a GSDL binding definition to connect to MODO services using protocols such as SOAP/HTTP, TCP/IP, etc
- Write a GSDL service definition based on the portTypes supported by the MODO and identified in first step
- Implement a factory by extending the FactorySkeleton provided indicating how new instances of MODO services are to be created
- Configure the factory with the options available

- Implement the functionality of the MODO service by extending the FactorySkeleton class. If an existing code (legacy code) is to be used in this mode, the *delegation* mechanism should be used, in this case the factory returns a skeleton instance
- Implement code for the client that must interact with the MODO service

2.8 Critical variables that make grid service practically useful

New technologies often face the challenges of being adopted by companies for certain reasons ranging from cultural, economic, political, operational, technological or even social in nature. Adoption of new technology by companies means some change, and ‘change’ naturally is challenging and difficult to take off smoothly, though necessary if companies must improve in efficiency and productivity. Grid computing is a new technology that aims to offer software functionalities as services to different disciplines. The critical variables that make grid services practically useful are classified into three categories-economic, technological and operational. This section intends to discuss these variables and incorporate them in the proposed service framework in this research. This is because the framework is intended to deliver practical services for real life MODO applications as demonstrated in the choice of the 3 case studies.

2.8.1 Economic variables

One of the incentives for companies sponsoring research project in grid computing is for economic reasons (Abbas, 2001). Companies naturally adopt new technology if it promises to improve return on investment (ROI), reduce total cost of ownership (TCO) of existing systems. For example companies do not need to own computation resources but can subscribe for the services and be charged per usage (Buyya, 2002). This takes off the cost of staffing and maintenance of the grid infrastructure. The concepts of grid services that are considered as economic variables for adopting grid technology are described below.

2.8.1.1 Putting wasted idle CPU to use: Optimum utilisation of existing computing Systems

Companies such as banking, oil and gas and aerospace companies invest huge sums of money and resources in purchasing and maintaining HPC systems for computational

and data intensive applications. In the course of industry survey, it was found that these companies already have grid departments for application and research activities. Most of the research is on managing the companies' data and metadata. Grid computing using Condor and Globus middleware can put to use idle processing power of desktops in a company to perform computational and data intensive tasks (Berman *et al.*, 2003). This is done through a method called 'cycle-stealing' which takes up processing power of desktops while not in use to where computation is done so that the computational speed can increase. The system releases the 'stolen' cycle time when the user of the desktop resumes by clicking the keyboard or mouse. In this way, companies will save huge sums of money meant for HPC and purchase of cooling and maintenance equipment for running HPC systems. This concept brings efficiency and reliability to the task of accessing huge datasets from distributed locations (Kosar and Balman, 2009). In one of the industry survey visits as part of this research activity, an oil and gas company believed that it will adopt grid computing if it can use all its desktops for simulation of seismic data without having to invest more on its existing HPC systems.

2.8.1.2 Subscription of services: Reduction in TCO of computing systems

To own and maintain all computing applications used by a company requires huge amount of resources in terms of manpower, software, hardware and maintenance cost. It is more economical and affordable to subscribe for some computing services and be charged on-demand basis. In this way, companies do not worry about personnel cost, maintenance cost, hardware cost and software license issues. This reduces the TCO of computing systems in terms of savings accrued from affordable grid service subscription. This is the concept behind the term 'grid' which was coined from electricity 'grid' as a service that is affordable, pervasive, reliable and cheap to use. Grid services providers use the concept of economics of scale to make their services cheap and affordable for grid service requestors.

2.8.1.3 Productivity: Reduction in time to market

In manufacturing, accelerated product development through agile manufacturing techniques which encourage concurrent engineering help to reduce the time to get a new product into the market (Zhao and Sobolewski, 2001). For example in life sciences, time to discover new drugs could be reduced by computational synergy of

using computational power and data storage as services. Grid problem solving environments such as FIPER (Federated Intelligent Product Environment) and SORCER (Service-Oriented Computing Environment) were developed to improve economic productivity in manufacturing industries through the provision of manufacturing tools as grid services.

2.8.2 Technological variables

Technological features play great role in making grid services useful. Technological variables that make grid services practically useful are protection of intellectual property and legacy systems of companies through secured and robust security features, scalability and adaptability.

2.8.2.1 Security: Protection of intellectual property and proprietary systems

One of the technological variables that will make grid services practically useful is if grid service requestors are assured of adequate protection of their intellectual assets such as data, information and software from intruders such as hackers and competitors. Any thing short of that might not encourage companies to participate in the grid service market. The single sign-in authorisation and authentication system in Globus toolkit is addressing such issues to make companies have confidence in the system (Berman *et al.*, 2003).

2.8.2.2 Scalability

Grid services should be able to accommodate extension when the company grows in its computing needs. Dynamic addition and removal of grid services is implemented in the open grid standard architecture platform. This is part of the effort to ensure scalability. Web services resource framework scales grid services to converge with web services. MODO applications require scalable platform that allows distributed resource sharing. In this research, scalability is assured by using Globus middleware which runs on open grid service architecture within the framework.

2.8.2.3 Adaptability

Service interface implementation in open grid service architecture platform interface, open grid service architecture platform model and open grid service architecture platform ensures that services adapt to dynamic changes in state instances and failures without affecting the performance of service quality (Berman *et al.*, 2003). Grid

systems manage heterogeneous distributed resources that are dynamic in nature. This calls for the concept of self adjustable and manageable interfaces in the grid infrastructure.

2.8.3 Operational variables

Operational variables concern end users' interaction with grid services. These include systems administrators and application programmers.

2.8.3.1 Multiple system domain administration

Grid services are administered from decentralised multiple administrative domains. This makes service provision complex from the human angle. The grid resource allocation management (GRAM) which is the main resource manager of grid services is needed in all grid nodes and so each node can manage its resources even when other nodes fail. In addition, the grid vision is that administration can be done by services themselves as explained in section 2.3.5.1, reducing human interference (Foster and Kesselman, 1999). This allows easy management of distributed and dynamic resources across administrative domains. In this research, different resources are located on different nodes and are managed by GRAM and scheduling is done using Condor scheduling system. The web monitoring and discovery service makes the resources transparent to all nodes.

2.8.3.2 Grid portal and programming interface

Grid portals allow common entry and access to grid services globally. This provides application programmers to develop services that can be accessed from different geographical locations around the world. Changes to applications can be implemented and replicated from any location. Grid security features for authentication and authorisation are part of the grid portal to ensure that only legal members are allowed access to grid resources. DECGrid uses the Globus security feature to register users and allow them access to nodes by specifying their identities in the map-file.

2.9 Facilities to develop MODO services

To be able to implement services, the research tried to study the facilities that have been used in literature. This subsection will concentrate on facilities required to build MODO services within grid environment. To build a reliable grid system, certain facilities are required. Major grid facilities for implementing grid services are

discussed below so that during design and implementation of DECGrid in chapters 5, 6 and 7, it will be easy to identify the components to be used.

2.9.1 Middleware

One of the main components of grid infrastructure is the middleware. Middleware is the software and hardware systems that enable interoperability among different platforms and users of the grid from distributed locations (Foster and Kesselman, 1999). For example, different users use different operating systems, communication protocols, databases, software, hardware, certificate authorities (CAs) and schedulers. For these different grid components to ‘talk’ to each other effectively, good middleware services are required. Globus Toolkit is one of the most popular grid middleware. MODO services need robust middleware to provide collaborative environment for professionals using different platforms.

2.9.2 Portals

Portals are the access points to grid resources. One of the efficient qualities of a grid service is the single sign-on method that allows a user to have access to any resource. Portals have GRAM (Grid Resource Allocation Management) job manager and gatekeeper that authenticate and authorise users before allowing them to have access to the grid. The proposed DECGrid uses portal for users to have access to resources that they have rights to access. This solves the problem of multiple authentication and access. However, this feature is sometimes criticised by industrial grid users (SurrIDGE and Taylor, 2005). Companies do not want third parties to give access to other users on their behalf. They want to have autonomy over the rights to share data and information. This might be for competitive business issues.

2.9.3 Certificate authority

Trust and security is the key to a successful grid deployment that provides MODO services (Foster *et al.*, 1998b). Managing cross-boundary trust among many users using different grid certificates requires the involvement of the certificate authorities. A certificate authority (CA) is a body or organisation that issues certificate for grid users. Examples of CAs are Globus and Science and Technology Facilities in the US and UK respectively. A certificate is the public key (encrypted) and information about its owner linked to the CA digital signature. Every resource, service or user has a certificate. A computing resource has a host certificate (hostcert) and host key

(hostkey). A user has a user certificate (usercert) and user key (userkey) and resource or service has resource certificate/key or service certificate/key. This pair (certificate and key) is called host credentials for computers, user credentials for users, resource credentials for resources and service credentials for services. There are public and private credentials. Proxy credentials are short-lived credentials. A user is identified by the certificate subject in the grid map file. A grid map file maps certificate subjects to local usernames. A CA signing policy is used to place strict compliance to the information one gives to a CA to bind to a key. There are different CA models. VOs find it difficult to manage different CAs and may soon adopt a common CA model (Johnston, 2003). The Globus Toolkit GSI uses Secured Shell (GSSH) for authenticating resources and services that need to be transferred from a distant site of a grid. CA defines both global and local authorisation and authentication of users and services to ensure a trustworthy grid infrastructure.

2.9.4 Schedulers

Schedulers ensure optimum utilisation of grid resources and services. Conventional scheduling systems such as Portable Batch Scheduling (PBS) and Sun Grid Engine (SGE) are used in conjunction with Globus GRAM to perform resource scheduling within VO. Condor is a dynamic scheduling software for the grid which works with GRAM to perform cycle stealing. Condor-G, as it is called, improves computing throughput by utilising the idle cycles of a company's desktops when users are not using their systems. This has the economic advantage of reducing the amount of money and resources spent on supercomputers (Abbas, 2001). Different grid users have different requirements and priorities. Schedulers ensure that these different users are satisfied through appropriate allocation of resources to them.

2.9.5 Resource brokers

Resource brokers receive request from users and sends these to GIS. A GIS search for the resources or services requested for and sends feedback to users through resource brokers. Resource brokers are the negotiating 'middle-men' between grid resources and users. Nimrod-G and Gridbus are good examples of resource brokers (Buyya, 2002). Resource brokering is an integral part of the grid infrastructure. It ensures feedback loop to users on whether or not a resource or service is available. This feature plays a role in ensuring QoS to users.

2.9.6 Operating systems

Operating systems (OS) such as Linux, Mac OS, Sun Solaris, UNIX, MS Compute Cluster Server and Windows OS play major roles in grid deployments. Linux happens to be the most widely used OS among grid users. This is because of the open source philosophy of Linux, and grid infrastructures are based on open source standards. Linux, UNIX and some operating systems already have in-built security and file transfer protocols that enable large-scale computing. For example, transferring large files from one machine to another machine at a distance using the command ‘scp <source file> <destination file>’ in Linux provides encryption for the file on transit until it reaches its destination. This complements the Globus GSI. In addition, OS ensures stability and reliability of grid systems. Some OSs have Multiple Processing Interface (MPI) built in them. This makes parallel scheduling easy as users can choose which processor they want to send their jobs to rather than just a random allocation of jobs to processors. OSs occupy low-level resource layer in the grid architecture, signifying their importance as resources for communication among different grid users from geographically separated sites (Foster and Kesselman, 1999).

2.10 Gaps in literature

From Figure 2.2, the research shows that most grid optimisation research uses the grid for computational and data intensive purposes. This is basically to speed up computation or simulation of data. Literature does not report the requirements from grid environment for MODO. Though projects such as the Geodise and DAME used grid for optimisation, there is an absence of high-level grid service specification document for designers to support MODO. This document should include features that bind service providers and end users. The document usually seen is the use case diagram and class diagram for programmers to develop the services. There is a need to include at the beginning of the service specification a service level agreement between providers and end users. This will ensure that grid services comply with some level of quality of services provided.

Another research gap is the lack of precise process for providing services. This includes steps such as identification of service users, identification of type of suitable grid services for the users, definition of service requirements for the users and identification of the need for service level agreement and service implementation. By

providing such steps, it will be clear as to which type of grid services are suitable for a particular application. Again, the service specifications found in literature are mainly for the use of programmers and not for other users of the system. This research provides service specifications for MODO so that it can be used and referenced by both programmers and users. The design of the service and schema for quantitative and qualitative mathematical model building in the MODO is another gap in literature. The proposed interface provides an easy interface for designers when carrying out optimisation.

Literature highlights the need to develop a grid architecture to support the MODO service. Grids are meant for large-scale distributed applications. Initial grid implementation issues for scalability and collaboration are most of the time neglected. A campus grid could eventually grow into a national grid and so there is the need to put in place measures that will take care of such issues. This research provides a step by step road map to tackle this issue. The steps are shown as the process that a service provider needs to follow at the initial stage of creating MODO services.

2.11 Summary

This chapter reviews literature in grid computing, grid projects, MODO methods and problem solving environments. Web and grid service protocols, middleware and scheduling systems were also reviewed. Gaps in literature were identified. The next chapter will use these findings to provide the aim, objectives, research questions and methodology for the research.

Chapter 3 - Research Aim, Objectives and Methodology

Having reviewed the literature and identified the gaps in the previous chapter, it is now important to identify the aim, objectives and methodology for the research. This chapter will concentrate on providing the aim and objectives for this research. It will also describe the methodology that will help in accomplishing the aim and objectives. The reasons for adopting the methodology and using 3 case studies to validate it will also be highlighted.

3.1 Aim of the research

The aim of the research is to develop the specifications and architecture of a grid service for multi-objective design optimisation.

The above aim shows that the research develops the specifications and architecture of a grid service that is meant for multi-objective design optimisation applications only. However, the problem solving environment has generic service specifications/functions that can perform generic optimisation in different fields of science, engineering and business. To accomplish this aim, the objectives in section 3.2 are proposed.

3.2 Objectives of the research

The objectives of the research are:

- *To understand the requirements from grid environment for multi-objective design optimisation(MODO)*
- *To develop functional grid service specifications for MODO*
- *To develop a grid architecture to support the MODO service*
- *To develop a step-by-step process that a MODO service provider needs to follow to implement the architecture*
- *To validate the research results using case studies*

The aim and objectives are used to guide the formulation and choice of a methodology.

3.3 Research methodology

The approach used to accomplish the aim and objectives of the research will be discussed under methodology. Figure 3.1 shows a summary of the methodology and the descriptions of the steps are in sections 3.3.1 to 3.3.5.

3.3.1 Literature review

The review of literature started with presentations on overview of grid computing, distributed computing, optimisation problem solving environments, e-Science projects and service-oriented architectures. The key-words were used to search for journal and conference papers. These research papers were then classified into computational grid, data grid, semantic/knowledge grid, problem solving environments and multi-objective grid-enabled optimisation. This classification forms the basis of the initial contribution to knowledge. This resulted in the categorisation of grid evolutions into generations of grid computing. The tools and variables that are important for developing grid environments were identified during this literature review from various publications on grid projects.

The service specifications of problem solving environments that were studied enabled the researcher to identify the requirements of grid services for multi-objective design optimisation. The PSEs studied are GEODISE, FIPER, SORCER and DAME and were discussed in section 2.6. This helped the researcher to also identify the gaps in the requirements of MODO. The first objective of the research is achieved through literature review and industry survey.

3.3.2 Industry survey

The next step is industry survey. 3 companies were visited. These are Schlumberger Oil and Gas Company in Oxford, UK, BAE Systems, Filton, UK and Microsoft Corporation, UK. The visits gave the research a practical view of what grid-enabled optimisation tools should deliver. About 35 online questionnaires (using QuestionPro software application) were sent to respondents and 1 telephone interview was also conducted to gather data. This enables the research to reach wider respondents from countries such as the USA, Germany, Denmark, France, UK and Nigeria. The

industry survey helped to crystallise the notion of production grid read from literature. It was evident from the industry survey that the middleware choice and security concerns seem to dominate the industrial application of grid resources.

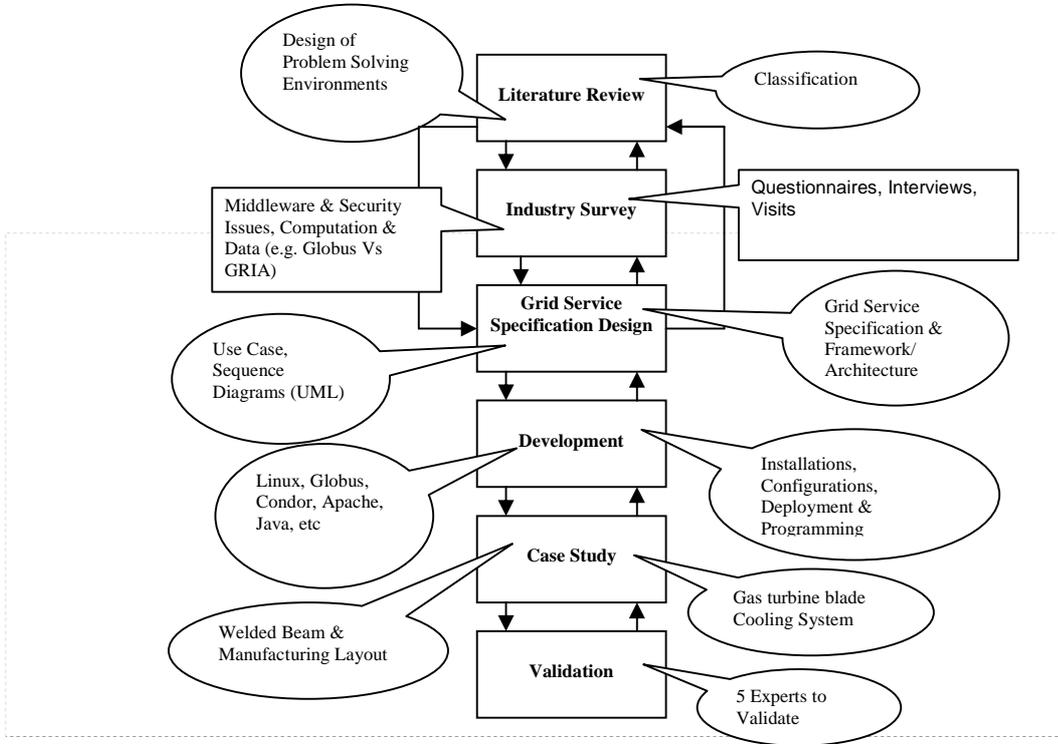


Figure 3.1: Summary of methodology

The industry survey further helped in understanding the grid requirements for MODO. The interfaces that were developed for MODO were in line with fulfilling the requirements for ease of use of the system by the designers.

3.3.3 Grid services specifications and design

The knowledge gathered from literature review and industry survey equipped the researcher to fashion out the analysis and design phase process. Unified Modelling Language (UML) diagrams were used for the object-oriented design using Rational Rose modelling tool. The design describes service specifications for compute service, collaboration service, mathematical building service and optimisation service. The specifications follow steps that a designer needs to follow to perform MODO using graphical input interfaces for entering design parameters.

The analysis and design used UML diagrams to model the service specifications identified in literature and industry survey. The functional specifications were

developed to achieve the second objective of the research. All the functional specifications and use case scenarios are presented in chapter 5.

3.3.4 Grid architecture for MODO service

The specifications developed in chapter 5 provided the building blocks to formulate a framework and architecture to support the specifications as services for MODO. The development involves using the framework to provide the services and the architecture to physically and practically implement the services using the tools identified in section 2.9 to support MODO applications. This enabled the researcher to accomplish the third objective in the research.

3.3.5 Implementation

The implementation is based on the framework and architecture described in section 3.3.4. The framework provides a step-by-step process for implementation of MODO service using the architecture developed. Globus Toolkit 4.0.4 and Condor were used in Linux operating system for the development of the prototype. 8 desktop computers form the DECGrid. Javascript is used to develop the client side and PHP is used for the server side. Apache is used as the web server. A quantitative (Q^T) math model creation service is developed to allow generic development of math models. This is a step by step grid-enabled service that allows distributed optimisation experts to collaborate and come up with a mathematical model. A qualitative (Q^L) model service can also be run on another node. Both Q^T and Q^L can be merged from yet another node to run the optimisation. The third case study on the design of a manufacturing plant layout enables a designer to visualise designs and rate them qualitatively in conjunction with the quantitative objective which minimises the cost of building the plant. The development adopts a factory pattern approach to provide services using java programming language. The implementation of the framework and architecture helped to achieve the fourth objective in the research.

3.3.6 Case study and validation

Three case studies were used to test and validate the prototype. The case studies are the optimisation of gas turbine blade cooling system, optimisation of welded beam design problem and optimisation of manufacturing plant layout/floor planning. The results were compared with the ones presented in literature. 5 experts validated the 3 case studies using various scenarios on how the system works for validation.

Extensive validation was done through practical demonstration of the software using these real-life case studies. The case studies are briefly described later in this section.

The implementation of the three real-life case studies is used to validate the framework and architecture developed. This accomplished the fifth objective. Various scenarios of running the system were used and experts filled in the questionnaires after participating in the scenarios during validation. The effectiveness of the framework using the specifications developed for MODO applications was identified by the experts.

3.3.6.1 Gas turbine blade cooling system

The first case study is the optimisation of gas turbine blade cooling system (Tiwari, 2001 and Roy, 1997). To enhance the performance and efficiency of gas turbine engine, the turbine blades need to operate in an environment where the gas temperatures are as high as practicable. However, this temperature sometimes exceeds the operational limits of the turbine blade materials. To maintain the integrity of the blade component and at the same time operate at high gas temperatures, the blade materials are cooled to safe operating temperature levels by passing cool air in them or over them in the form of films. A small part of the compressor exit airflow is used to cool the blades. However, the temperature of this cooling air depends on the compressor pressure ratio and on the flight Mach number and temperature. The trade-off for the blade cooling includes loss of work and efficiency due to the portion of the air taken from the compressor exit. Hence this problem can be described as a multi-objective problem with four objectives namely coolant mass flow for radial passage, coolant mass flow for film hole, metal temperature for gas side and metal temperature for film side. It has 12 variables. The objective is to increase the efficiency of the turbine engine by maximising the temperatures and minimising the coolant mass and still maintaining the integrity of the blade.

3.3.6.2 The Welded beam problem

The second case study is the welded beam design problem. This problem describes how a beam needs to be welded on another beam and must carry a certain load. The objective of the design is to minimise the cost of fabrication and minimise the end deflection (Deb, 2001). Here, the overhang portion of the beam and the applied force are specified, thus making the cross-sectional dimensions of the beam and the welded

dimensions as the variables. The problem has four constraints. The first constraint is to make sure that the shear stress at the support location of the beam is smaller than the allowable shear strength of the material. The second constraint is to make sure that normal stress at the support location is smaller than the allowable yield strength of the material. The third constraint is to ensure that the thickness of the beam is not smaller than the weld thickness and the fourth constraint ensures that the allowable buckling load of the beam is more than the applied load.

3.3.6.3 The Design of a manufacturing plant layout/floor planning

The third case study is the optimisation of a manufacturing plant layout (Brintrup, 2007) and floor planning (Sushil, 1993). This case study, unlike the first 2 which have only quantitative models, has both quantitative and qualitative models that need to be optimised. In addition, the case study has two separate models that need to be optimised at the same time using shared grid resources. The first model (design of manufacturing plant layout) considers the optimisation of the arrangement of various spaces in a manufacturing plant. The idea is to maximise the subjective expert satisfaction that concentrates on efficiency of the plant (qualitative model) and to minimise the cost of building the plant (quantitative model). The building is partitioned into stores, offices, assembly, warehouse, press room, paint room and spares. The problem with the design of manufacturing plant layout is that different factories have different considerations and so different tools are used to optimise the layout. The optimisation search space should explore possibilities that can ease understanding of the search domain objectives and incorporate expert opinion.

The second model is similar to the first. However, this time around the model is to optimise the floor planning of the area where possibly will house the staffers of the manufacturing plant. This problem finds the length and width of each room that will minimise the cost of the apartment in the presents of some constraints and maximise the comfort and efficiency of the apartment. Like the first model, the cost of each room is proportional to the area of the room except for the cost of kitchen and bathroom whose costs are two times their respective areas. This case study will demonstrate how grid resources will be shared by distributed experts to perform multiple model optimisation that have quantitative and qualitative objectives.

In all the three case studies, this research will use the developed grid environment to publish the objective functions, constraints and variables as resources and distributed experts will make inputs from the different nodes to collaborate. There is an interface that allows parameters to be entered and the model is linked to NSGA-II. The NSGA-II then runs the optimisation. The results obtained are compared with the results that were obtained in literature using other environments. The flexibility of sharing data during collaboration is also validated.

3.4 Reasons for adopting the above methodology

This section explains the reasons that necessitated the use of the methodology presented above. First, to understand the current state of problem solving environments that has been used to solve MODO and related problems, the researcher opted to read the pioneering book in grid (Foster and Kesselman, 1999) and some later similar books (Abbas, 2001) and (Berman *et al.*, 2003). These books exposed the researcher to the initial concept of the grid through the later stages of evolution of the grid to a service-oriented problem solver (Foster *et al.*, 2002a). Later findings from conference and journal papers showed an increase in grid research projects sponsored by governments and companies from many countries (Hey and Trefethen, 2002). What led to the classification of the grid evolution are the changes in vision and new concepts that are constantly witnessed in successive grid projects. De Roure *et al.* (2003) classified this evolution as first, second and third generations. This research extended the classification to the fourth and future generations of grid computing. An important achievement in literature review is the identification of MODO challenges and requirements which are discussed in sections 2.5.2 and 2.6. Section 2.5.2 described MODO challenges and its special features and section 2.6 helped the researcher to understand the requirements for MODO applications by studying related problem solving environments and their challenges in solving MODO problems.

The need for industry survey arose as companies and governments sponsoring grid research projects for real and tangible economic benefits of the grid projects (Abbas, 2001). This is because early research in grid was concentrated on building middleware, protocols and scheduling systems (Krauter *et al.*, 2002). Another reason for the industry survey was to find out the problems and challenges in deploying grid systems so that the researcher could learn from them. A visit to the companies proved

to be essential to the research as issues such as grid security, middleware choice, support and training and even the choice of which application area and operating system to be used for deployment of grid systems featured. Details of the industry survey are provided in the next chapter.

The literature review (chapter 2) and industry survey (chapter 4) enabled the researcher to come up with the functional grid service specifications that suit both academic and industrial contexts. The high-level description of link between literature review, industry survey, design of service specification, service framework and implementation of proposed architecture is shown in Figure 3.2. The description highlights the sequence of major achievements in the research and the respective chapters that described them.

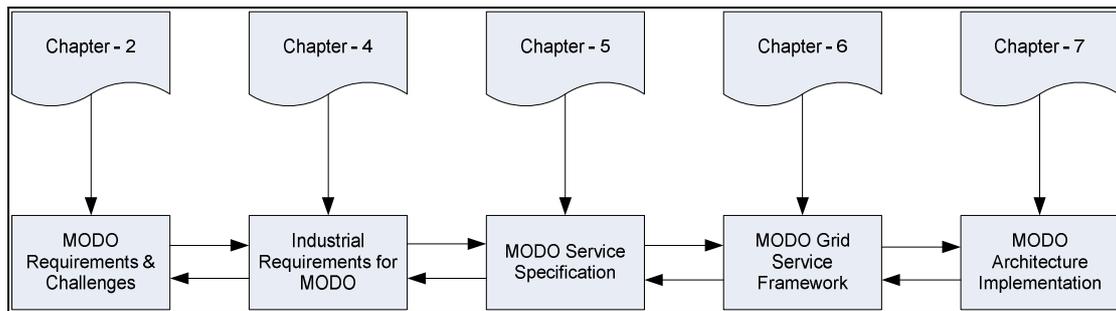


Figure 3.2: Important sequence of research achievements and chapters

The development of the grid architecture commenced after analysing and designing the functional and non-functional requirements for MODO applications. Using the knowledge obtained from studying the architecture of web and grid services, the researcher developed an architecture that uses the master-worker model which allows providers to publish resources and run optimisation functions. To ensure scalability and provide a robust system, a step-by-step process is identified for service providers to follow during implementation. This process includes identification of grid users and their requirements which help in the choice of grid to be implemented. Inclusion of service level agreement which specifies the responsibilities of providers and users ensures compliance to quality of service. It is important to use real-life case studies so that the results could be verified against those already established in literature. Three case studies were chosen from aerospace, manufacturing and mechanical engineering.

Details of the choice are discussed in chapters 4 and 8. The final stage is to validate the prototype using the case studies and results obtained.

3.5 Summary

This chapter provides the direction for the research by highlighting the aim, objectives and methodology. The reasons for adopting the methodology proposed are discussed and 3 case studies to be used for validation have also been briefly described. The next chapter discusses the findings from industry survey.

Chapter 4 - Grid in Industry

The last three chapters have given the research the basis and theoretical background of grid services in multi-objective design optimisation. This chapter intends to use these findings and reconcile them with what is expected of a ‘production’ grid in industry. The chapter is also tailored towards achieving the aim and objectives of the research in industrial context. Personal visits and interviews, questionnaires and online-survey methods of gathering data were used in the industry survey.

4.1 Industry survey

The concept of grid computing is appealing to companies that require huge computational and data resources to process data for their day-to-day business transactions. Companies also need collaborative tools to share virtualised information with their partners on-demand basis (Goteng *et al.*, 2008). Researchers, scientists, engineers and companies that work to improve the performance of products using design optimisation techniques also require tremendous amount of computational resources. An industry survey is conducted to identify grid application areas that are most required by companies and the associated limitations and problems faced by them in implementing such applications. Application areas covered in the survey include multi-objective design optimisation (MODO), collaboration, resource sharing, data storage and computation in aerospace industries, oil and gas companies, research centres and software development companies. The study covered application areas other than MODO because they are required by MODO experts. For example, MODO experts need to share resources and collaborate to take decisions on designs. To give the survey some generalisation different areas such as oil and gas and aerospace applications are included.

4.1.1 Aim and objectives of the survey

The aim of the industry survey is to give the research an industrial context by finding out what the business community expects from the emerging grid technology and compare this with academic efforts with the view of identifying the gap (if any) between industrial and academic grid solutions. The objectives of the industry survey are:

- *To validate the findings in literature and to identify the gap between research and real applications using grid.*
- *To identify if the companies are aware of the grid technology.*
- *To have a first hand information on what companies are looking for and what business problems they expect the grid technology to solve for them.*
- *To identify migration, integration, cost and other aspects of grid implementation issues.*
- *To understand the fears companies have about grid technology.*

4.1.2 Industry survey methodology

Four methods of gathering data were adopted in the course of the industry survey. These are face-to-face interview, telephone interview, on-line questionnaire approach and observation. The reasons for using each method are discussed.

For example, face-to-face interview was adopted in two companies that were visited because the personnel were available and the researcher used the opportunity to observe the simulation system in parallel with the interview. The first company visited had an oil and gas exploration simulation systems and the second design optimisation system for aerospace company.

In case of the telephone interview conducted for an executive officer in a company, the respondent had very busy schedules and suggested a telephone interview. The interview went well and besides it had the advantage of saving time and resources.

The online questionnaire was adopted to reach out to researchers and companies within UK and other countries. This proved to be very useful as respondents came from many countries. The reason is to give the research an international perspective.

The combination of these methods gives a rich information gathering opportunity to the researcher. The industry survey approach, methods used and the companies visited are discussed below.

4.1.2.1 Choice of companies and respondents

The first step in the industry survey was to identify the companies that are likely to be interested in the research. This was obtained from literature and the companies' needs for huge computational and data capabilities. Other features that determined the selection of companies are the nature of information flow and collaboration, multidisciplinary nature and multi-criteria design nature of their products. After choosing the companies, only the departments relevant to the research were targeted. This is to ensure that the researcher deals with staff that have knowledge or are likely to understand the subject area of the research. 1 company from oil and gas, 1 company from aerospace and 1 company from software industry were identified for visits and interviews. Respondents from research centres and other disciplines were reached at through online-questionnaire. Table 4.1 shows the companies/institutions visited and the criteria used to select them.

Table 4.1: Companies/institutions visited during industry survey and criteria for selection

Company	Task	Criteria for choosing companies					
		Computationally intensive	Data intensive	Multi-objective	Multi-disciplinary	Collaborative	Distributed
Schlumberger oil company, Oxfordshire, UK	Seismic data simulation	✓	✓	✓	✓	✓	✓
	Decision making				✓	✓	
	Exploration optimisation	✓	✓	✓	✓	✓	✓
BAE Systems, Filton, UK	Design optimisation	✓	✓	✓	✓	✓	✓
	Decision making				✓	✓	✓
	Trust and contract management (TrustCoM) project	✓	✓	✓	✓	✓	✓
	Grid Resource for Industrial Application (GRIA) project			✓	✓	✓	✓
Microsoft UK	Commercial platforms (operating system & application)	✓	✓	✓	✓	✓	✓
	Commercial middleware development			✓	✓	✓	✓
Department of Trade and Industry (DTI)-Intellect, UK	Creating awareness in grid computing			✓	✓	✓	✓
National e-Science Centre (NeSC) Edinburgh	Science & engineering research	✓	✓	✓	✓	✓	✓
	Middleware			✓	✓	✓	✓
	FireGrid project	✓	✓	✓	✓	✓	✓
	Metadata project	✓	✓	✓	✓	✓	✓
	Nano complementary metal oxide semiconductor (NanoCMOS) project	✓	✓	✓	✓	✓	✓

4.1.2.2 Design of questionnaire

A combination of closed and open-ended questions was adopted. There are more closed questions as they are normally easier to be administered and get responses, though the researcher was mindful of their drawback of not allowing other options that might come up. This drawback is compensated in the questionnaire by providing a field for comments/other options in each question. The open-ended questions were few and deliberately made short and require short answers. This is to encourage the respondent to continue filling the questionnaire. There were two sets of questionnaires to serve interviews and online survey respectively. However, the contents are similar such that the analysis of the results was integrated easily. The differences were in the presentation. For example, the questionnaire used for interviews has more details and consist of more open questions. This is because the researcher could easily ask to clarify questions that are not clear to the respondents. The online questionnaire has more short and straight questions so as not to waste respondents' time as the researcher is not there to clarify certain questions. The questionnaire for interviews was further made easier to suit the telephone interview. The telephone questionnaire has short and straight answers that do not require lengthy explanations. See Appendix-I for samples of the questionnaires.

4.1.2.3 Approach to respondents

The researcher made efforts to establish contacts with the heads of the departments intended to be visited by attending the Cranfield University career fare where most of the companies were present. This was very successful as the researcher was encouraged to visit the companies. E-mail addresses and telephone numbers obtained from business cards of the officials served as means of first communication. The researcher followed up with the career fare activities and asked to be introduced to the appropriate personnel to administer questionnaire or discuss schedules for interviews. Before each visit, a brief description of the agenda and time that is required was sent 2 weeks ahead. This is to prepare the respondents in advance.

Face-to-face interview and observation

Two company visits were made in which a combination of face-to-face interview and observation were used. The companies are Schlumberger Oil and Gas Company (Schlumberger Information Solutions) and BAE Systems (Advanced Technology

Centre). In each company, the researcher gave presentations on the background of the research and its applications in that industry. The companies also gave presentations on their research projects that are related to web and grid applications after which the interviews commenced.

Telephone interview

As mentioned earlier, a telephone interview was conducted for one company. The company is Microsoft UK. This was a compromise for the inability to go to the company due to tight schedules of the Microsoft Officer. It however discussed very important areas that affect grid implementation issues at both low-level and high-level applications. The discussions were centred on operating system and applications. Details of the discussions are presented in section 4.1.2.4.

Internet-based questionnaire

A web-based online questionnaire was also used to gather data. The software used is called QuestionPro (www.questionpro.com). The questionnaire administered manually was converted to an online questionnaire with little modifications on the presentation making the questions shorter and less open-ended. The researcher registered with organisations such as the IEEE Computer Society, British Computer Society, JISCMAIL and other Grid research organisations. The online questionnaire was sent to members of these organisations. There were 21 responses, 5 dropped out after starting to fill in the questionnaire, 66 people only viewed the questionnaire and a completion rate of 80.77% was achieved. These figures were generated automatically by the QuestionPro package. This approach proved to be very successful as respondents from different countries such as USA, France, Germany, Denmark, Nigeria and UK gave interesting responses. The respondents were from research centres, universities and commercial companies. This demonstrated the power and usefulness of online questionnaires to conveniently reach out to many respondents across the world and get feedback promptly. However, the researcher was also mindful of the limitations of administering online questionnaires. Such limitations include bandwidth delay in opening e-mails, lack of time and interest in continuing with long questionnaire and long ambiguous questions that require clarifications. To avoid these problems, the questions were closed-structured and require short answers with an option for comments. The software has good statistical

and analysis tools for data representation. The online questionnaire allowed the research to create different types of questions ranging from fixed to open-ended and single to multiple answer types.

Figure 4.1 shows the process of sending the questionnaire to respondents with an introductory e-mail stating the purpose, time it will take to fill the questionnaire and promise of confidentiality in analysing the data provided by respondents.

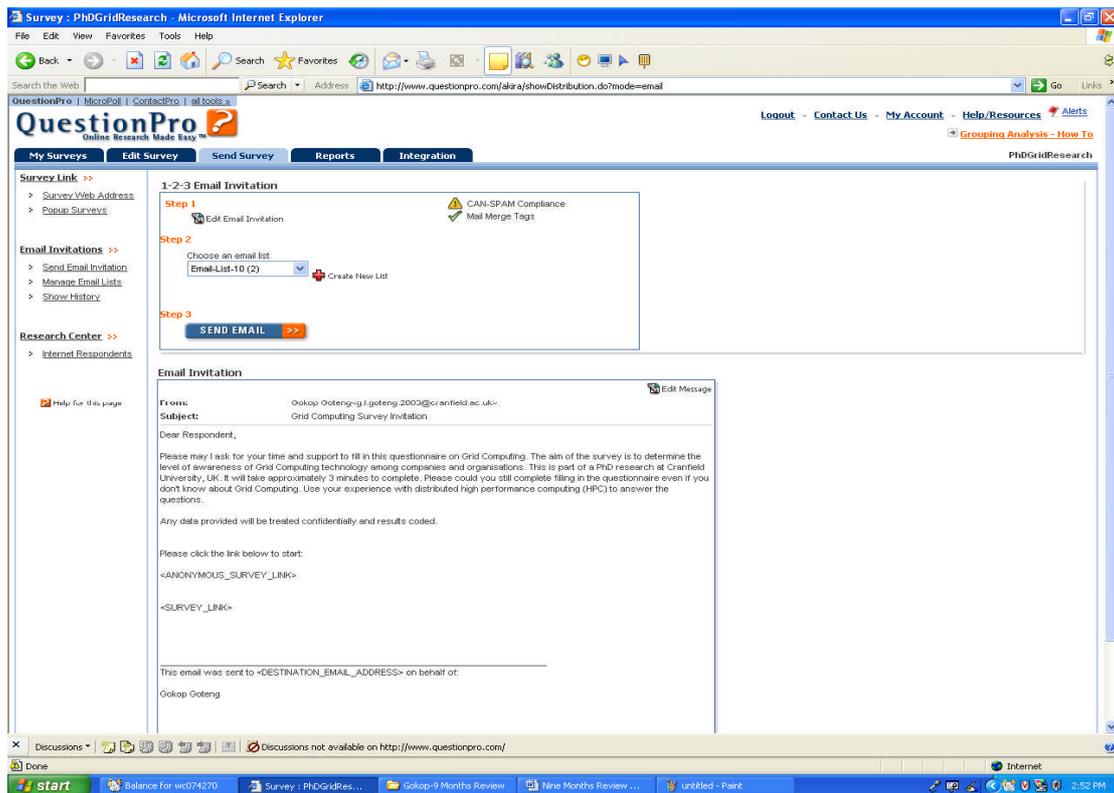


Figure 4.1: Introductory e-mail message for the online questionnaire

4.1.2.4 Companies that were interviewed

The three companies interviewed are discussed below. These companies are Schlumber Oil and Gas Company, BAE Systems and Microsoft-all three in UK.

Schlumberger Oil and Gas Company: Schlumberger Information Solutions (SIS)

About 12 personnel of SIS UK were given presentation and two of them were interviewed by the researcher on the 6th of December, 2006. The respondents included senior and technical staff members who have high performance computing experiences. One of them is in charge of grid technology research. The researcher

filled questionnaire as the interactive session of interview started. SIS has developed software called Eclipse which is used for the simulation of seismic data during exploration. The system is run on cluster of high performance computing platform. A demo of the system was given by the head of SIS who had about 15 years experience explained as the researcher observed and asked questions on the functionalities of the software. The software demonstrated that oil and gas simulation is a computational and data intensive field and the company appreciated any additional improvement on computing power. Oil and gas reservoir simulation is also a multidisciplinary field which requires the collaboration of geologists, petrochemical engineers, economists, political scientists, sociologists and environmental scientists. Grid laboratories can provide a virtual organisation for these experts to collaborate to make informed decisions. Seismic data analysis is data intensive and computationally expensive. Data grid is another aspect through which the research can help the oil and gas sector in managing distributed seismic data across the company. An area of challenge here is that SIS has not implemented grid and is not thinking of implementing it at the time of this interview. However, the respondents agreed that if grid applications are developed and tested to prove that it can improve computational power, the company will be ready to start thinking of implementing grid. The manager in charge of SIS led the team of SIS in the discussion. The manager responded to the questionnaire on behalf of the 12 staff. The meeting ended with the observation of the reservoir simulation systems developed by SIS. The system does computational calculations on seismic data and analyses the data to forecast oil and gas reserves for a particular region. Decisions are taken based on the results of the simulation runs. Optimisation process is also performed by the system and graphical display of the seismic analysis results exists. The system is developed using Fortran and C++. The system is run on Linux Cluster of 24 processors and is on the desktops of the SIS staff.

BAE Systems: Grid and Web Services

Two staff of BAE Systems at Filton, UK were interviewed on the 22nd of January, 2007. The head of Web Services and Grid-based Services who is a doctoral degree holder and has about 17 years of experience and the System Technologist who holds a masters degree in computing gave the researcher 3 presentations on what BAE Systems is working on in the areas of web and grid services. There is a project called TrustCoM (Trust and Contract Management) which developed software to serve as

the proof of concept for collaboration to ensure trust, security and contract management among companies using web services. Another project called the GRIA (Grid Resource for Industrial Application) developed a grid middleware that implements grid authorisation and authentication based on bipartite algorithm not on single-sign-on proxy. Bipartite algorithm ensures that only resource owners give access to users and not through any third party while the single sign-on proxy system allows users access to grid system only once and gives access to them through resource owners as well as through third parties. This is the basic difference between Globus security feature and that of GRIA. At the end of the presentations, the researcher presented the research topic, aim and objectives to the team. The researcher then proceeded with the interview and filled in the questionnaire. BAE Systems is working for the UK defence industry and is cautious of sharing its resources and knowledge with any other company. This perhaps informed its decision to develop the GRIA middleware where authorisation can only come from the owner of the resource and not by proxy. BAE Systems believes that grid will play a great role in shaping a new infrastructure for large-scale science. The issues raised focussed on bridging the gap between academic research concepts of the grid and industries' expectation of the grid. This can be done through collaboration between researchers and companies. The Globus toolkit is described as addressing academic issues rather than real world application issues.

Microsoft UK: Low level grid development platforms

A telephone interview was carried out with the Microsoft UK on the 12th of January, 2007. One personnel in Microsoft Technology Office UK who holds a masters degree and is a chartered professional with 20 years experience was interviewed which lasted for about 30 minutes. There were only 10 questions, bearing in mind the tight schedule the respondent had. These questions as explained earlier represented a summary of the questionnaire used for the face-to-face interviews at SIS and BAE Systems. As said earlier, the sample of this questionnaire can be seen in Appendix-I. From the interview, it is gathered that Microsoft is positive that grid will emerge as a new infrastructure for large-scale computing in the future. Microsoft is engaged in developing low-level systems to serve as platforms for grid application programmers. This explains why Microsoft launched its first Microsoft Compute Cluster in 2006. To explain the late arrival of its super computing cluster, Microsoft said it is after mass

production and so the supercomputing technology is becoming more affordable and so many companies and individuals will start to use clusters for various applications. Grid can only be embraced when grid platforms are steady like windows and web-based programs. The Microsoft Compute Cluster is not expensive and researchers can afford to pay for and have more intuitive and graphical interfaces that other present supercomputing operating systems lack. Grid security and scheduling systems are incorporated in the new compute cluster. From the interview, Microsoft intends to provide standardised compute cluster that will provide graphical user interface for submitting and monitoring jobs without the need to issue commands from the command prompt as done in open standard system such as Linux. This according to the respondent will include good documentations and support facility. Microsoft has released HPC (High Performance Computing) Compute Cluster 2008 to serve as an operating system for HPC and grid communities.

4.2 DTI-Intellect: Grid Computing Now! Industry internship

A six-week industry placement was sponsored by Intellect also known as Grid Computing Now! (a division in the UK Department of Trade and Industry (DTI)). Intellect is responsible for creating awareness and promoting the use and application of grid computing in industries. This opportunity arose as a result of the researcher winning the UK grid computing competition organised by Intellect and British Computer Society and sponsored by Microsoft. Before starting the internship, the researcher wrote a proposal of places to be visited and activities to be engaged in during the six weeks. A meeting with the Managing Director, Technology Manager and the researcher was arranged and modalities, aim and objectives of the internship were agreed upon. See Appendix-IV as part of the terms of reference of the internship.

The aim of the internship is:

To enable the researcher to meet grid users in research centres and companies and have hands-on experience on grid technology to understand issues surrounding grid deployments.

The first one week was spent at the National e-Science Centre (NeSC) Edinburgh, Scotland. At NeSC, the researcher was introduced to the middleware and training

groups by a senior research representative of Intellect at NeSC. The middleware group used gLite (Lightweight Middleware for Grid Computing), NGS2 (National Grid Service 2) and Globus middleware. The researcher learnt there that a common problem with all grid middleware is the challenge in resolving cyclic dependencies in production grids with regard to dynamic resource discovery during collaboration. The Rapid Project is another work in NeSC that aimed to provide rapid development of web and grid portals for users. This project developed portals for users to submit jobs and allow multidisciplinary experts to exchange metadata was used to support this cyclic dependency. The Metadata Resource Model Catalogue is another research area at NeSC which provide methodology on how distributed experts can use data and metadata efficiently. This concept was described and it was explained how it can provide a generic model for all kinds of metadata. For this reason, it is advisable to store only references of metadata instead of the metadata. In this way, a lot of storage space is saved and a generic way of accessing the metadata is possible. Two models are used for resource discovery namely push and pull models. Push model checks the life of the service while pull model sends request to host to enquire if the service is alive. This is also used to implement service-level agreements to ensure quality of grid services.

The researcher also had discussions with the training group at NeSC. The training group provides training for users in institutions and companies. Globus 2.5 is the main middleware used for training. Implementation issues and configuration of services were the main focus of discussion with the training team. It was maintained that all grid deployments should take the issue of support and training seriously to sustain grid projects. The researcher met with the team that worked on the Fire Grid and NanoCMOS (Nano Complementary Metal Oxide Semiconductor) projects and obtained insights on the projects. The Fire Grid project is a demonstrator that shows the power of remote execution of resources such as simulation models and building structures to alert Fire Brigade to respond to fire outbreaks promptly. It is a system which uses sensors to receive signals and analyse data for decision making. Nano-CMOS project demonstrates how databases and information can be shared in a distributed manner among distributed users that are working on designing new nano devices for companies. This project used GridSAM (Grid Job Submission and Monitoring Web Services) for job submissions to ensure efficiency of processing.

GridSAM is a scheduling and monitoring system that ensures computational throughput for jobs. Two people from NeSC were interviewed on the last day of the visit.

The remaining time of the internship was spent on having meetings with different business and technical experts in grid technology. The experts are from Department of Trade and Industry (DTI), Barclays Capital Grid Department and Platform Computing. The need to shift from research to production grids was constantly emphasised.

4.3 Industry survey findings

Literature reveals that the trend of research activities tend to focus on meeting industry needs as grid researchers moved from middleware related development to prototype development of grid systems to deliver business benefits based on quality of service (Abramson *et al.*, 2002). Also, the industry is willing to work closely with researchers to identify grid features that need adjustments to fit into a business tool such as making better use of their existing IT (information technology) systems. This section discusses the findings from research and industry through the industry survey conducted by the researcher.

The graphs presented in the analysis of the survey findings are based on the number of respondents from the companies interviewed plus the ones from online questionnaire. The total number of respondents is 38. 12 respondents were from SIS, 2 from BAE Systems, 1 from Microsoft, 2 from National e-Science Centre, Edinburgh and 21 from online questionnaire. Majority of online respondents are academics (about 18) with few from companies.

4.3.1 Grid applications issues

Like any new technology, companies are looking for practical applications of grid to improve upon existing systems. The first part of the questionnaire in this survey focused on why companies want to or intend to use grid computing.

Figure 4.2 is the analysis of figures obtained on why companies want to or intend to adopt grid computing. 32% of respondents said that their companies want to adopt

grid to improve efficiency. From the face-to-face interviews that the researcher had with respondents, efficiency was suggested to include speed of computation and simulation processes, better collaboration, decentralised data management and decision making, knowledge sharing and reuse. For example in oil and gas reservoir simulation and modelling, the goal is to generate both good estimates of reservoir parameters and reliable predictions of oil and gas production to optimise return on investment (ROI) by searching through large space of seismic data using grid computing infrastructure (Parashar *et al.*, 2005a). 24% of respondents are of the view that grid computing will bring reduction in the cost of their IT systems by maximising their usage. This they believed may be possible through the integration of systems and efficient utilisation of computational power using Globus and Condor-G to use cycle times of the companies' desktops instead of investing in buying expensive supercomputers. Another 21% of respondents believed companies need grid for competitive advantage. It is believed that since grid is aimed at solving large-scale problems that currently have stretched the limits of high performance computing (HPC) paradigm, new ways of tackling computational problems need to be introduced. These new ways may have market-oriented and economic implication for companies that are ahead in adoption of grid economic models (Buyya *et al.*, 2001). However, only about 8% of respondents believed that grid adoption might reduce time to market in their companies.

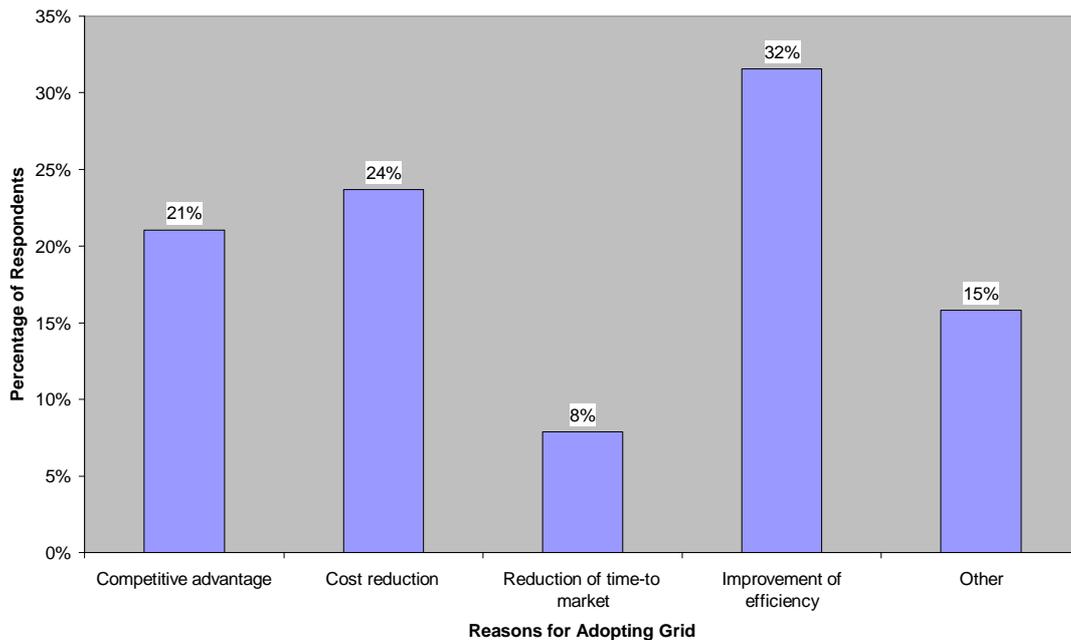


Figure 4.2: Grid computing incentives

It might be thought by many researchers that any technology that improves speed of data processing and analysis ultimately improves decision making and might help in getting new products quicker into the market so as to get a better share of the market. About 15% of respondents made other comments. In these comments, some said grid computing provides a new scientific computing paradigm upon which new capabilities or discoveries can be realised. For example in life sciences, drug discovery time can be reduced through optimisation of molecular modelling using grid computing (Sudholt *et al.*, 2005). Some other respondents said that they want to explore grid to meet customer's requirements or they are developing grid middleware because they got the funding to develop grid software for research purposes.

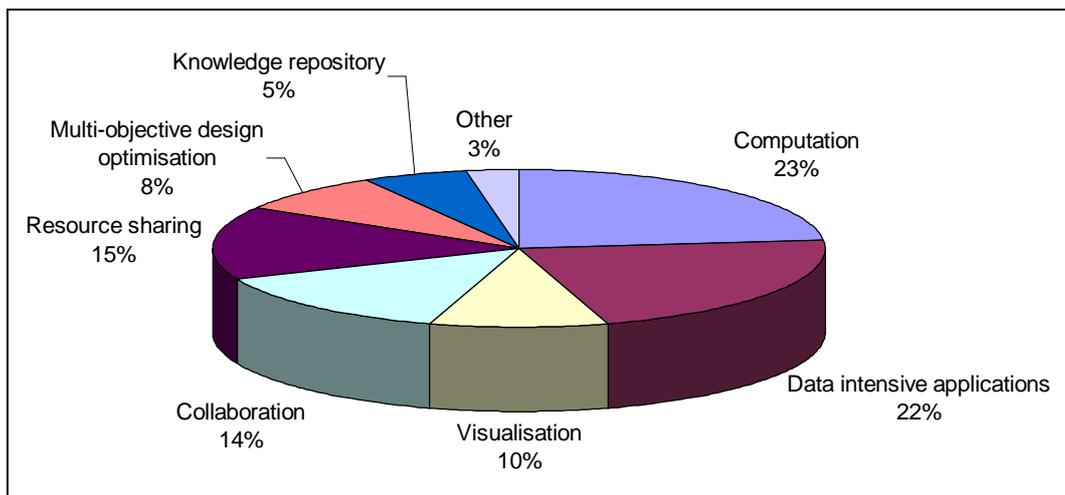


Figure 4.3: Grid application features and components

Figure 4.3 shows a pie chart with different grid components and features and percentages of responses against each one. Expectedly, majority (23%) of the respondents said that their companies or institutions used grid computing for computation. Data intensive application came next to computation as 22% of the respondents said that they use grid for data related applications. This further confirms the concept of grid computing based on large-scale computationally and data intensive problem solving infrastructure. It might also be a confirmation that cluster computing and related high performance systems are attractive platforms among researchers and academia to improve computational power and distributed data

analysis (Buyya *et al.*, 2002). Resource sharing, collaboration, visualisation and multi-objective design optimisation (MODO) application features got 15%, 14%, 10% and 8% respectively. Other comments such as using grid for remote control of instrumentation and scalable access to storage resources got 3%.

Figure 4.4 is a bar chart showing some grid application fields and percentages of respondents against each field. Traditional scientific disciplines which are known to be computational and data demanding took the lead as can be in the chart. 27% of the respondents said that they want to use or are using grid computing in physics related disciplines. This is followed by engineering design optimisation (EDO) which has 15% of respondents. In particle physics and MODO, computational fluid dynamics (CFD) and finite element analysis (FEA) tools are used to analyse the properties of materials to generate parametric geometries (Eres *et al.*, 2005). This perhaps is the reason why physics and EDO took the lead from respondents. Other disciplines such as general science and service-oriented applications both got 13%.

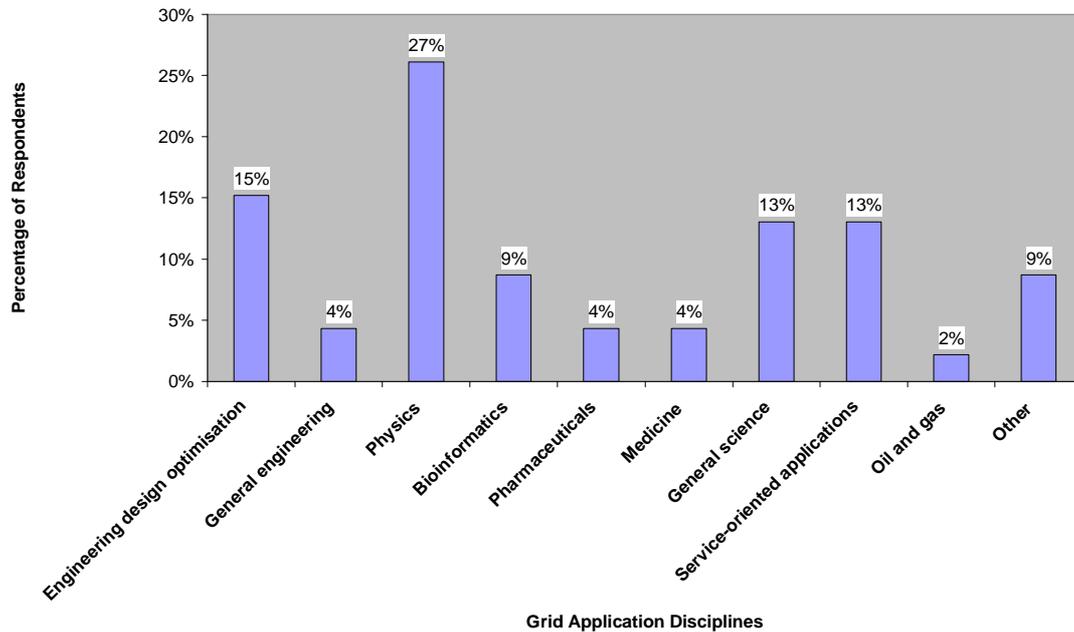


Figure 4.4: Grid application disciplines

Bioinformatics, pharmaceuticals and medicine got 9%, 4% and 4% respectively. Life sciences are looking for more computational power to discover new ways of diagnosis and drug discovery methods through molecular optimisation processes (Sudholt *et al.*, 2005). Only about 2% of respondents said that their companies intend to use or are

using grid computing in the oil and gas industry. Oil and gas reservoir exploration requires computational modelling of seismic data for optimum oil well production and utilisation. Other (9%) respondents believed grid is being used for social science, atmospheric science, enterprise application integration and chemistry.

4.3.2 Grid implementation issues

Implementation issues in grid computing projects are usually related to middleware choice, problem solving environments, operating system and open standard architecture.

4.3.2.1 Middleware

Figure 4.5 shows that Globus middleware is the most popularly used platform according to this survey as 48% of respondents said that they used Globus toolkit for their institutions' grid infrastructure. Globus is one of the early grid middleware which is widely used by researchers and academia. The popularity of Globus is due to its open source standard architecture and is also freely downloadable from the Globus website.

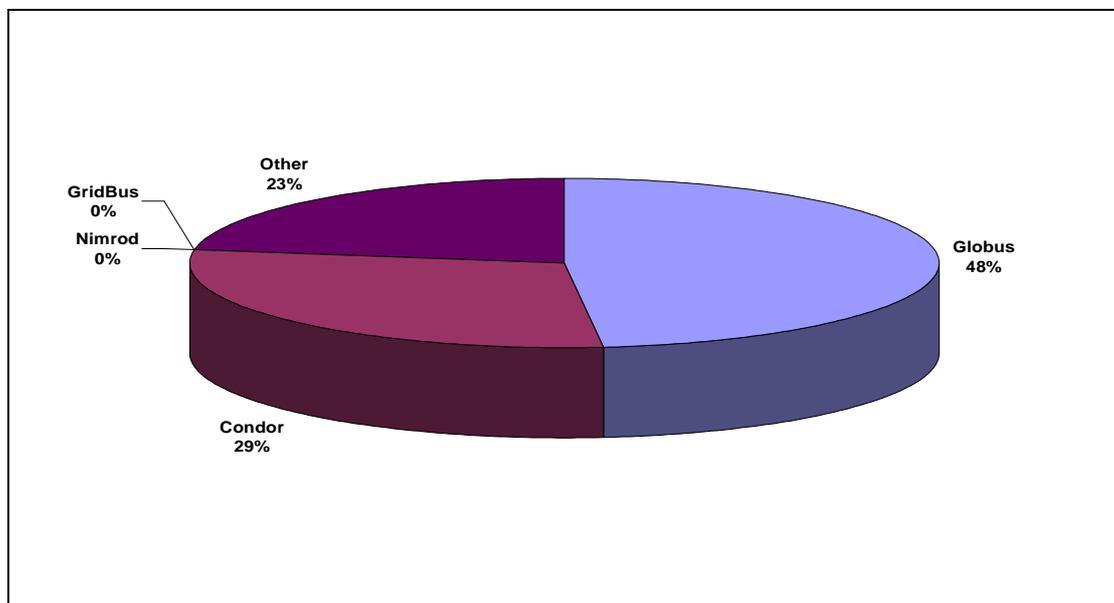


Figure 4.5: Grid middleware

29% of respondents said that they are using Condor (scheduling system) throughput computing. Condor is usually used with Globus Resource Allocation Management (GRAM) to have a Condor flavour called Condor-G (Condor-GRAM) to achieve

efficient utilisation of compute cycle times of a company’s computing processing power through cycle-stealing. 23% of the respondents said that they are using other middleware such as GRIA (Grid Resource for Industrial Application) and grid portals. No respondent indicated that his or her company is using Nimrod and Gridbus middleware. Nimrod is a resource broker that helps in allocating resources based on service rules and requirements while Gridbus is a middleware used for economic computational resource management research.

4.3.2.2 Problem solving environments

Globus toolkit is both a middleware and a problem solving environment. In Figure 4.6, it is again dominating as 35% of respondents said that their companies used Globus as a problem solving environment. This proportion constitutes mostly of the researchers in academia. Companies prefer to develop and use proprietary problem solving environments (PSEs). This category of PSEs falls under the Other option which came next to Globus with 31%.

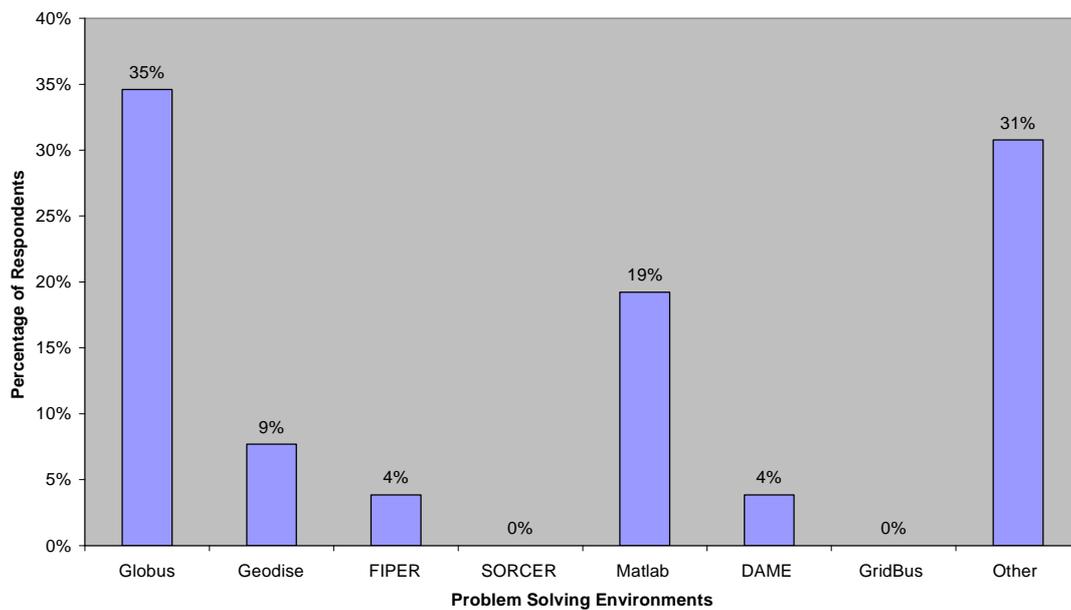


Figure 4.6: Grid problem solving environments (FIPER=Federated Intelligent Problem Environment; SORCER=Service-Oriented Computing Environment; DAME=Distributed Aircraft Maintenance Environment)

Such proprietary PSEs include GRIA, Virtual Data System and ModelCentre. Some of the reasons given by companies to develop custom made PSEs to conform to their business needs and to avoid the complexity in using Globus proxy security features.

Globus proxy security system allows third parties to give access to users on behalf of resource owners. 19% of users said that they used Matlab as PSEs. This again is connected with engineering design and academic research activities. Matlab has features for visualisation, fuzzy logic and neural networks which are often used by researchers in the intelligent systems community. Geodise, FIPER and DAME got 9%, 4% and 4% respectively. Geodise is a popular UK e-Science project which is used as a demonstrator for EDO.

4.3.2.3 Operating systems and open standard architecture

One of the crucial issues in high performance computing (HPC) community is the choice of which operating system (OS) to use. Scheduling large-scale computing processes requires robust OS that will ensure robust results in scalability, robustness, reliability, transparent location, open standard distributed storage and secured communication protocols (Iamnitchi and Talia, 2005). Figure 4.7 shows that 63% of respondents used Linux OS to deploy grid technology for their companies or institutions. The reasons given by respondents for using Linux include experience and standard OS for HPC. It is surprising that no respondent said that he or she is using Linux because it is free.

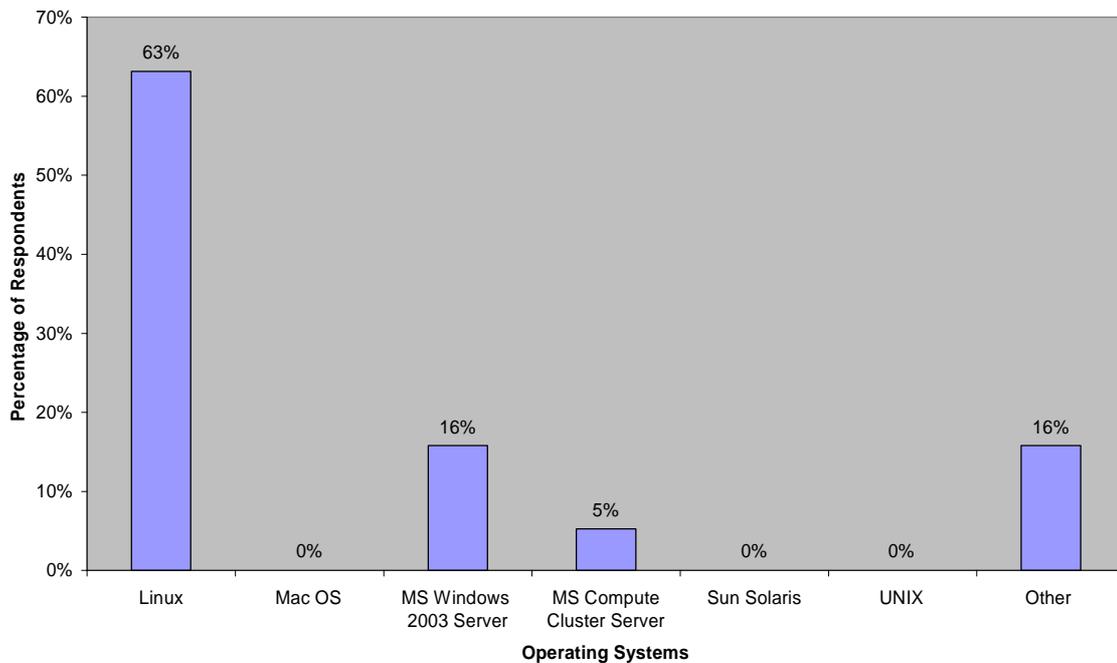


Figure 4.7: Operating systems for high performance computing

Windows 2003 Server and other OSs got 16% each. The surprising feature of this survey is that the new entrant, Microsoft Compute Cluster Server got 5%. MS Compute Cluster Server was introduced in 2006 to serve as a HPC Microsoft platform. With this success within a year's launch, this OS may play a role within the HPC community in the coming years. Though Mac OS, Sun Solaris and UNIX are important OSs among the HPC community, no respondent said he or she is using them. Some major problems pointed out by respondents about HPC OSs are lack of compatibility between different versions and parallel images and installation/configuration problems.

4.3.2.4 Limitations and challenges

There are some limitations and challenges in grid implementation. Some problems highlighted by respondents are interoperability, shortage of skills and migration, and integration of software and hardware. In Figure 4.8, 37% of respondents said that the challenge facing grid implementation is the interoperability problem in software and hardware platforms among collaborating organisations. This is a confirmation of the problem faced during the first generation grid computing projects that tried to develop grids for the science and engineering community by linking several supercomputing centres to demonstrate the synergy of computing power (Aloisio and Talia, 2002). 28% respondents said that the grid technology is new and there is shortage of skills in implementation due to the lack of experience. 26% of respondents said migrating and integrating existing systems within a grid virtual organisation is challenging, especially when it involves legacy and proprietary systems. However, respondents do not believe that cost is a problem in implementing grid projects. This might be true because most grid projects are funded by government agencies and there is no expectation on return on investments and as such no pay back analyses are being done to justify investments. In general, designing a grid architecture that will meet dynamic grid users' requirements is challenging due to the issues such as supporting adaptability, extensibility, scalability and different administrative domains maintaining site autonomy (Krauter *et al.*, 2002).

In a specific question, the respondents were asked to choose the most challenging aspect of grid service involving service providers and service requestors. In figure 4.9, 48% said that the security and trust related issues are the most challenging. Grid

computing allows resources including large-scale expensive instruments and data such as genetic database and sensitive business data to be shared among partners (Chadwick, 2005). To allow this, the organisations need to be assured that the system allows them to determine who accesses what and to what extent.

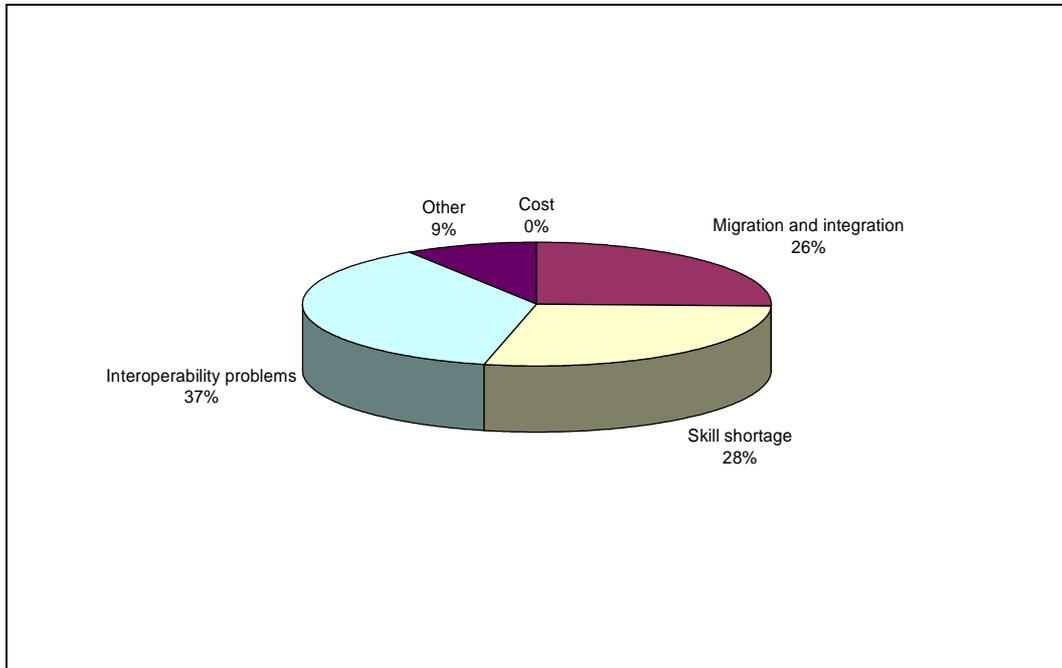


Figure 4.8: Limitations and challenges of grid implementation

Figure 4.9 shows that 5% of respondents believed that policies and rules are part of the challenges in grid services. These policies again are about restrictions to who is allowed to use what and to what extent. This confirms the complexity involved in the development of Globus Security Infrastructure (GSI) which adopts the single sign-on by proxy method. This is more appealing to the academia than industry. Industry does not like the idea of a third party giving authorisation to a user on behalf of the resource owner. This is why some companies prefer to use the bipartite authorisation method which restricts authentication and authorisation responsibility to the resource owner only. For example, the GRIA (Grid Resource for Industrial Application) project uses bipartite system for this reason.

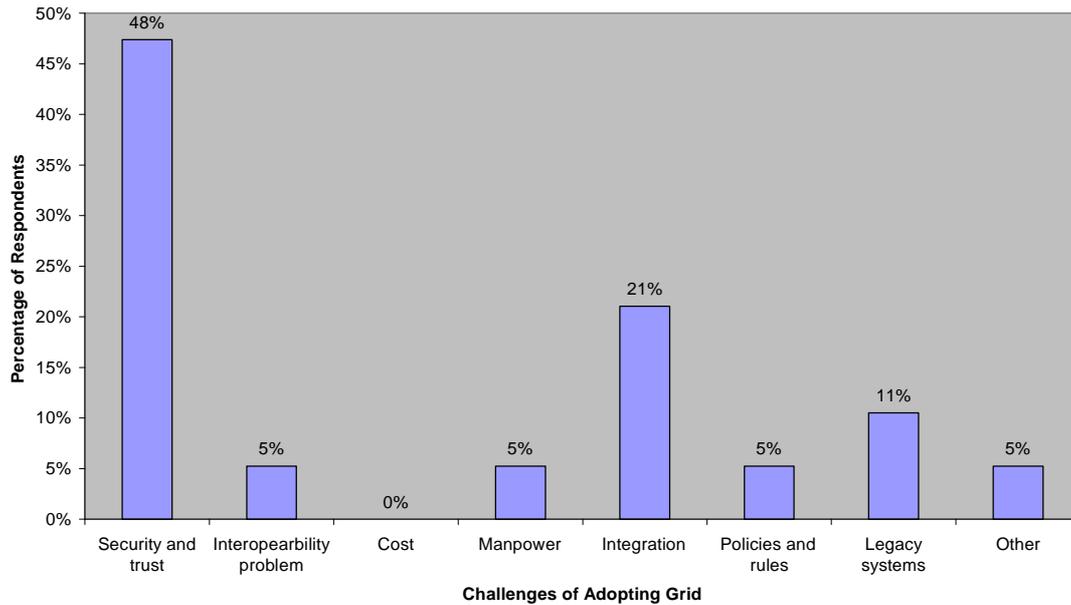


Figure 4.9: Challenges of grid implementation

4.4 Gaps between academia and industry

As with most new technologies, researchers often battle with issues that will make the technology practically useful to end users. This is also often used as a justification for continued funding from governments or companies. From the industry survey and the literature reviewed, it is evident that grid computing is in a stage that companies want to see grid applications delivering business benefits after over a decade of middleware, resource broker and problem solving environment research projects in science and engineering (Hey and Trefethen, 2002).

Table 4.2 summarises the opinion expressed by academia and by industry regarding some key features of grid. These opinions are the gaps between research and industrial applications of grid.

The research used the above findings to tackle the issues that are within the scope of the research. The design and implementation of services for computation, data and collaboration as described in chapters 5, 6 and 7 to provide solutions to the issues of computation/data, collaboration/sharing, scale/granularity and implementation. However, security, middleware/resource brokering, service platforms and hosting

environments are beyond the scope of this research. This can be considered for future research. This research only uses the available middleware (Globus) with its security features, service platforms and hosting environments facilities to implement services.

Table 4.2: Academic and industry opinions on grid features

Grid feature	Academic opinion	Industry opinion	Discussion
Security	Authentication and authorisation through proxy and delegation	Authentication and authorisation through bipartite system which restricts authentication and authorisation to owner of a resource	The Grid Security Infrastructure (GSI) should have multiple security features that serve these 2 authentication and authorisation protocols. Presently Globus is based on single-sign-on proxy which academia is comfortable with but the industry prefers to develop in-house security features that restrict the right of authentication and authorisation to the owner of the resource. The GRIA project aims at solving this problem for industry
Computation/ data	Parallel distributed based on proxy and delegation for both computation and data	Parallel distributed based on proxy and delegation for computation but bipartite system on data	Companies would not mind how computation is done but would want to restrict the right of who accesses their data to themselves, while academics restrict it through proxy because it reduces bureaucracy and makes it flexible for collaboration in research. The present system can be improved using Condor for cycle-stealing to optimise the utilisation of idle computing power for companies. This research suggests that both proxy and bipartite methods should be included in data grid implementation but relaxed in the case of computation
Hosting environment	Hosting environments consider scientific and engineering rules	Hosting environments consider business rules	Researchers should consider business rules first and then scientific/engineering rules
Collaboration / sharing	Open standard	Both open and proprietary standards	Collaboration among academia is based on open standard architecture while companies would want to develop their proprietary collaboration tools for sensitive applications. Globus implementation does not seem to take into account the fire-wall implementations of different companies for collaboration. Proprietary fire wall should be considered during collaboration
Services and platforms	Open standard systems run on Linux environment	Combination of open standard and proprietary standard run on Linux or a graphical user interface (GUI) operating system	Services are published and used through open standard protocols within academia but some companies would prefer services to be published and used through both open standard and proprietary standards taking into account the company rules and regulations. We suggest both open and proprietary standards to allow flexibility of choice

Grid feature	Academic opinion	Industry opinion	Discussion
Middleware/ resource brokering	Interoperability of hardware and software interfaces are the most important	Interoperability of systems, accounting, Payment metering and service quality are considered important to industry	Researchers should pay attention to accounting, payment metering on-demand and quality of services for resource brokering
Scale and granularity	Large-scale science and engineering disciplines with emphasis on computation and data intensive applications.	Large-scale enterprise integration with emphasis on data mining and collaboration among distributed customers and partners	Grid system can accommodate both large-scale science and enterprise collaboration between businesses
Implementation	Considers improvement in computational power	Consider business benefits first	Researchers should consider impact of grid implementation on business considerations. This is the only way grid technology can be adopted by companies

4.5 How literature and industry survey guided the design and implementation

The common findings in literature review and industry survey guided the design and implementation requirements. For example, Figures 2.3 (literature review) and 4.5 (industry survey) showed that Globus toolkit is the most popular middleware among researchers and industry. Figure 4.7 shows that Linux is popular among most application areas. This is why the research decided to use Globus within Linux environment to conform to the requirements of the grid community. Figure 2.2 in literature showed that majority (46 papers) of user application is for data related and Figure 4.3 in industry survey also showed that majority (22%) of respondents said they used grid for data related applications. This is why the researcher adopted PostgreSQL database server to accommodate secure sharing of data and information in the collaboration service. The idea to design a graphical user interface for creating mathematical model and carrying out optimisation is also obtained from literature and industry survey. The absence of features for MODO in literature and industry survey also influenced the framework of the research. For example, literature did not report a

structured process to define a grid service for MODO that guides the user through the process of mathematical model building. Thus the research implemented this to fill the gap. This implementation is guided by the aim and objectives which were also obtained identified from both literature and industry survey. The researcher studied service specifications of related problem solving environments (see sections 2.6 and 5.2) in literature and had discussions with industry experts during the internship at Department of Trade and Industry (see Appendix-IV). This is the basis upon which the service specification described in Figure 5.24 came about.

4.6 Summary

This chapter described the methods and findings of the industry survey. It compares these findings with literature and identifies the gaps between grid in research and industry. These findings in literature and industry survey are used to guide the design and formulation of the framework and architecture for implementation.

Chapter 5 - Grid Service Specifications and Design for MODO

The previous chapter highlighted the gaps between academic and industrial applications of grid services. This chapter intends to use these research and industrial issues to provide generic service specifications for MODO applications. The chapter presents the conceptual design of DECGrid interfaces and functionalities.

5.1 MODO service design concept

Software systems that offer functionalities as services do so by allowing organisations (providers) to expose the service capabilities programmatically over grid-based or internet-based infrastructures using conventional standard extensible mark-up language (XML) based programming language and protocols that are SOA-compliant (Papazoglou, 2003). These services are implemented through self-describing interfaces (portTypes) as described in chapter 2 using service data elements (SDEs) and operations. Before the emergence of grid services, the applications service provider (ASP) model was the pioneering third party system that allowed deployment, hosting and management of access to applications for distributed customers as services (Walsh, 2003). This model brought about the economy of scale needed by companies for their business automation needs as ASP's sole responsibility is to maintain the applications and entire infrastructure and ensure availability of data to distributed users when needed (Papazoglou, 2003). ASP model became the de facto infrastructure for outsourcing software capabilities as services across the world. The grid provides such capabilities with much larger and more scalable capability using the open grid standard architecture (OGSA) and web service resource framework (WSRF) technologies and protocols (Foster and Kesselman, 1999). The PSEs described in section 2.5 of chapter 2 use the concept of grid services to provide services using the programming XML-based schema that use grid service description language (GSDL). These PSEs as mentioned before do not provide structured service specification documentation for end users where they can see clearly the systems' capabilities that can serve for service level agreement negotiations of the services along side the XML-based service specifications. This research is proposing a MODO design concept in which structured process of delivering services to MODO experts

are contained in a service specification document in conjunction with the XML-based schema for programmers. This gives the grid-based system a utility perspective that puts in the minds of programmers the customers' needs and requirements and allow for service level agreements. In this research, ASP is not used but an equivalent open source language known as PHP is used with Java to accomplish what ASP can do.

The researcher considered users of the system in capturing requirements to come up with MODO design. A user in this context is defined as a designer who is engaged in MODO and needs distributed computational resources for collaboration with other designers to be more efficient. Two categories of users are considered. These are research and industry users. Requirements for the users are captured in literature and industry survey. In literature (section 2.6), problem solving environments (PSEs) were studied. The service specifications of these PSEs are also treated in section 5.2. These PSEs provide functionalities for optimisation. The researcher also studied the working principle of MODO and how this relates to these PSEs. During industry survey, research and industry users were asked to mention the incentive of using grid services for MODO (see Figure 4.2). Majority (32%) of respondents said they want to improve efficiency. By efficiency from the comment option in the questionnaire, they mean easy process of using the service, graphical user interface and access to computational resources to collaborate with other designers. These reasons from literature and industry survey led to the designs adopted in this chapter.

In view of the above description, the aim of this chapter is to:

Provide the designs for the MODO grid services

The objectives of the chapter are:

- *To describe and compare the designs of different models that provide MODO services*
- *To use the designs to come up with the framework and architecture for the services*
- *To provide a structured specification document for end service users*
- *To use the designs as a road map that will guide the implementation of the services*
- *To come up with designs for DECGrid MODO services*

The unified modelling language (UML) will be used for the designs. This is because UML is easy to be understood both by programmers and non-programmers. The Rational Rose modelling software package will be used to produce the UML diagrams. Rational software has capabilities to convert designs to programming language templates and this efficient capability is exploited in this research.

5.1.1 Specifications of MODO service

Four services that interact to accomplish the MODO task of building a mathematical model to carry out optimisation are designed. The services are mathematical model building service, parameter input/NSGA-II optimisation service, computational service and collaboration service. The details of how these services are coordinated and work together to accomplish MODO tasks is discussed under each design. The implementation of MODO services is preceded by a specification document. This document contains the service interfaces, operations, areas of application and optimisation strategies to be used. This section discusses the different specification models and dimensions of service specifications that are used to come up with a model for this research in the next chapter. Service Specification Strategy (SSS) for the study is proposed using the Web Monitoring and Discovery Service (WebMDS) in Globus Toolkit. WebMDS serves as the interface to view all optimisation resources and to send queries for subscription. This research is proposing a Multi-Objective Design Optimisation Search Strategy (MODO-SS) which assumes that there is a model but the model can be continuously refined based on the inputs of multidisciplinary design engineers tailored towards satisfying some design criteria and constraints by following the step by step guide provided by the prototype developed. This makes the engineers who do not have knowledge of optimisation to perform optimisation as they work through the variables, constraints, objectives and parameter input interfaces.

5.1.2 Different models to deliver MODO services

It is important to briefly describe some models that are used for MODO services so that the research can provide its own model and compare its features to the ones provided by them. Different service providers adopt different models to specify services. The FIPER (Federated Intelligent Product Environment) and SORCER (Service-Oriented Computing Environment) problem solving environments adopt a

model that enables programmers to have a federation of networked objects to implement remote service-oriented interfaces (Sobolewski, 2007). The model uses the unified modelling language (UML) notations for specifying classes, methods and use cases. Other models are Geodise, GRIA (Grid Resource for Industrial application), DAME (Distributed Aircraft Maintenance Environment) and Globus. The Geodise model specifies services using the open grid standard architecture specification that interfaces web services. Computational specifications are based on the Condor pool method. Geodise uses ontology for workflow specifications to drive the design search using knowledge based approach. Matlab is used for visualisation within Geodise framework. The GRIA middleware restricts the right of optimisation resource ownership to the owner. This is the model that industries prefer to use over Globus model which is widely used by researchers. The DAME model collects data on the fly and uses it for maintenance purpose. This study uses Globus middleware and Condor scheduling system to implement MODO service capabilities and services that were mentioned above. The WebMDS in Globus is customised to serve as the area where optimisation resources can be viewed by distributed optimisation users. Queries can be sent to filter the desired resources and services. The Condor system schedules computational resources among the nodes for optimum sharing and utilisation.

5.1.3 Dimensions of a service

The first case study in this research as briefly described in section 3.4.5.1 of chapter 3 has 4 objectives, 4 constraints and 12 variables. The computation of these objectives is interdependent on one another and this creates more computational challenges. The services that are proposed for MODO in this research take into consideration the dimensionality of MODO problems with regards to objectives, constraints and variables (Brockhoff and Zitler, 2006). For example, the third case study on the design of a manufacturing plant layout/floor planning has multiple models (2 models) with 1 quantitative objective which is computed from core NSGA-II mathematical model file written in C programming language and the other objective is a qualitative objective obtained from a different application written in C++. The qualitative objective is merged with the quantitative objective to perform the selection operation, crossover and the process continues for successive generations. The heterogeneity in objectives and computational platforms adds to the complexity of dimensionality. The grid infrastructure allows PSEs to incorporate these heterogeneous resources with

ease using the self-organising and scalable implementation of grid index information service and grid service handle that uniquely manage each service instance (Andrzejak and Xu, 2002). This research uses this feature of the grid to accommodate dimensionality and heterogeneity in MODO problems. In addition, dimensions of service specifications cover functional and non-functional requirements. Functional dimension specification includes multi-job submission and requests to different domains, computational synergy to speed up processes and resource-sharing component optimisation. Non-functional service dimension specification consists of quality of service (QoS) implementation. QoS includes service reliability, fault tolerance, availability and timing. The Globus middleware specification schema has these built-in capabilities which this research used to specify the time and availability of services and fault report summary.

5.1.4 Quality of MODO service

Although quality of service (QoS) is not part of this research, users of any service may require some level of features that satisfy their needs for using that service. This is part of the reasons this research wants to propose the service specification document which shows users the expected deliverables of the services and also an opportunity to reach a service level agreement. For efficient sharing of technical data and files through scheduling and co-allocation to multiple administrative domains, QoS issues are important in multidisciplinary domains (Foster *et al.*, 2001). The quality of MODO services is measured in terms of availability, reliability, interoperability, scalability, integrity and transparency. Availability means the users can have access as well as use grid resources and services whenever the need arises. Users should be able to terminate or increase the time of usage. Reliability means grid services are stable and perform well even in the presence of errors or faults. This is important in multi-objective optimisation where many experts collaborate and need to share resources and expertise. Although this research does not provide this capability, the web monitoring and discovery service is used by collaborators to view and ensure that resources are available before making an attempt to use them. Different resources such as data and objective functions are viewed and shared among users. The design of the prototype allows different grid nodes to serve as both servers and clients to ensure scalability and efficient MODO collaboration. Transferring huge optimisation data across the nodes in secured manner is facilitated by the GridFTP (Grid File

Transfer Protocol). Future middleware research is looking at incorporating agent-based QoS to configure dynamically the service level agreements as computational resource availability changes are unpredictable due to the simultaneous use of the resources by distributed users (Al-Ali *et al.*, 2004).

5.2 Analysis of service specifications

This section discusses the important service specification features of 4 problem solving environments. This is to prepare the research on how related works have delivered services to users. Service specification is one unique way in which different problem solving environments present the functionalities of the services they offer. A service specification document is a high level presentation of the classes, objects, methods and properties that make up the services. A UML (Unified Modelling Language) is usually used to present this document. This is an important document during service development and implementation. Before proposing the different stages of this research's design for its service specifications, it is important to briefly compare and contrast 4 different service specifications. This consists of Globus, FIPER, SORCER and Geodise service specifications. These are not the only problem solving environments (PSEs) or middleware that have service specifications. There are also service specifications for DAME, GRIA, Gridbus, Nimrod, Cactus and Condor. However, this research will only consider Globus, FIPER, SORCER and Geodise as they represent a general view of service specifications that this research is focusing on.

The analysis is carried out by studying the functionalities of the PSEs. Since grid systems are based on layered architecture, the analysis considers the layered arrangement of each PSE to provide services. The UML diagrams are either obtained from reference sources in literature or drawn by the researcher using information from literature and industry survey. For example the SORCER service specification was obtained from Sobolewski (2007) when the researcher asked for it through an e-mail and obtained permission to use it while the Globus service specification is drawn based on information obtained at Globus website (www.globus.org), literature and industry survey. The UML diagrams consist of the classes describing the operations and properties of the services offered by the PSEs. The services are all based on the open grid service architecture and web services resource framework.

5.2.1 Globus service specifications

The Globus Toolkit is in its version 4.2.1 at the time of this research. Globus is modular and its service specifications are based on the different service modules. The open source nature of Globus works well with other service oriented architectures (SOAs) and incorporates the open grid service architecture (OGSA), web services resource framework (WSRF) and simple object access protocol (SOAP) protocols in its service specifications. Globus adopts a layered service specification that implements different services in different layers as explained in section 2.6.5 of chapter 2. For example, core network services are implemented at the communication layer while the application layer implements user application specifications. Specifications in Globus are in form of interfaces known as PortTypes. Figure 5.1 shows a high level service specification document of Globus.

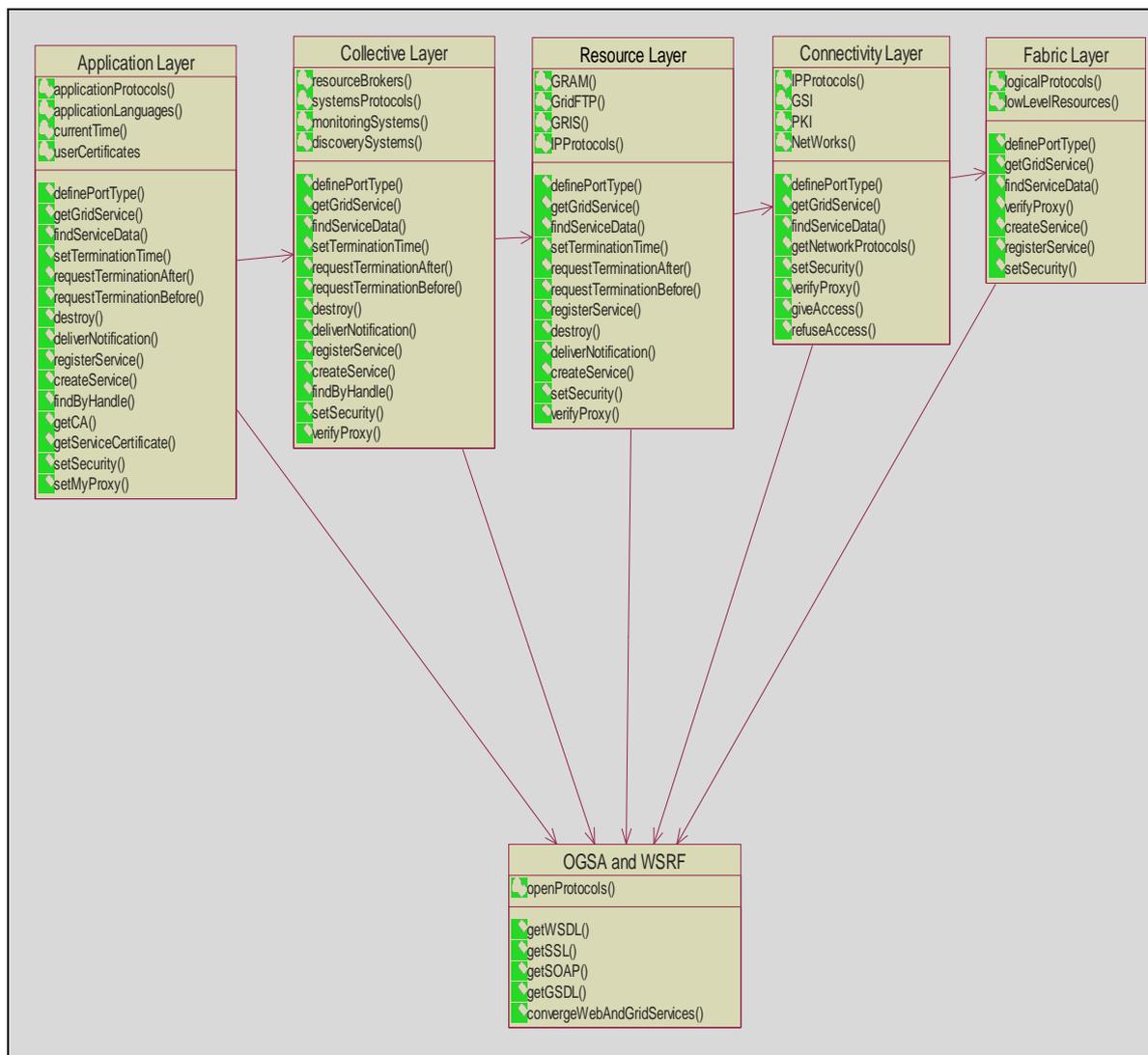


Figure 5.1: Globus service specifications

Each layer needs a GridService portType. This is because the GridService portType is the fundamental interface upon which all service instances act as grid services (Amin *et al.*, 2002). Each layer implements services using operations such as createService(), findServiceData(), setTerminationTime() and many more. Globus specification has the advantage of publishing services in the Web Monitoring and Discovery Service (WebMDS) within the Index Service so that services are transparent to the entire virtual organisation (VO). In this research, computational resources and algorithms are published in the WebMDS so that optimisation experts may view these resources before actually using them. The Globus service specifications deal with the software functionalities of the services.

5.2.2 FIPER service specifications

FIPER has service specification for concurrent engineering applications. Its specifications are tailored for object-oriented programmers to use and implement capabilities that effectively distribute and manage companies' valuable product development assets. These assets include data, models, applications and computational resources. Grid computing and web technologies are used as enabling infrastructures for running FIPER within multi-institutional enterprises. FIPER is based on web-centric architecture which has presentation layer, business logic layer and repository layer (Zhao and Sobolewski, 2001). The presentation layer is the specification for the HTTP portal where clients have access to FIPER application services. The business logic layer hosts the application server which contains service providers and has capability for service monitoring and discovery. The repository layer consists of databases which are shared concurrently by users. FIPER is used mostly for concurrent engineering applications in manufacturing.

FIPER and SORCER both use the conventional 3 basic components of SOA (Service Oriented Architecture) namely service provider, service requestor and service registry for publication, execution and discovery of services and resources. In SORCER, service providers are specified as network objects with states, behaviours and types within the service specification document. Network requests are called exertions and are carried out by requestors. Figure 5.2 shows the service specification document of SORCER. This document describes the Federated Method Invocation (FMI) of service interfaces, classes, sub-classes, tasks, jobs and methods (Sobolewski, 2007).

This document is a class diagram using UML notations. SORCER is used mainly to expose functionalities to programmers that are developing services for concurrent simulation and optimisation processes.

5.2.3 SORCER service specifications

The SORCER service specification document addresses programmer’s problems more than business rules and service level agreements. This is one advantage that Globus has over SORCER. Globus specification incorporates WSRF rules which take into consideration present web service specification with business rules such as meeting QoS agreements.

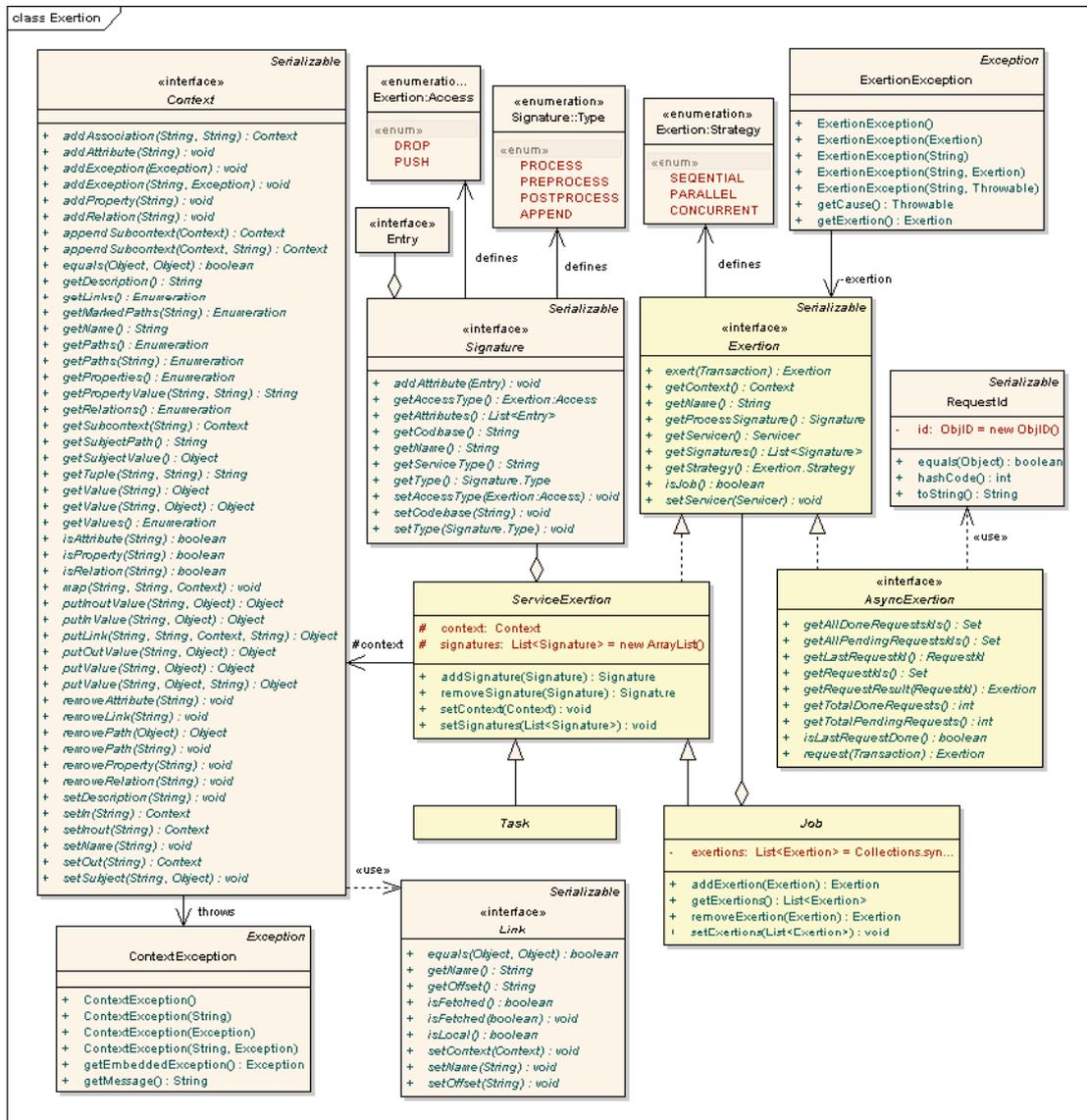


Figure 5.2: SORCER service specification document (Sobolewski, 2007)

5.2.4 Geodise service specifications

Geodise uses service specifications that are specific for engineering design optimisation. However, it incorporates some generic tools from Globus Toolkit and Matlab. It also uses ontology to describe designs for different experts. Geodise uses functions to perform operations within its computation and database components. The functions start with the prefix `gd` which stands for Geodise and followed by a hyphen (-) then the operation's name. For example the function `gd_archive` is a database function that stores a file in a repository and its associated metadata. Figure 5.3 shows service specifications of Geodise. Geodise is also open source software that can be downloaded free.

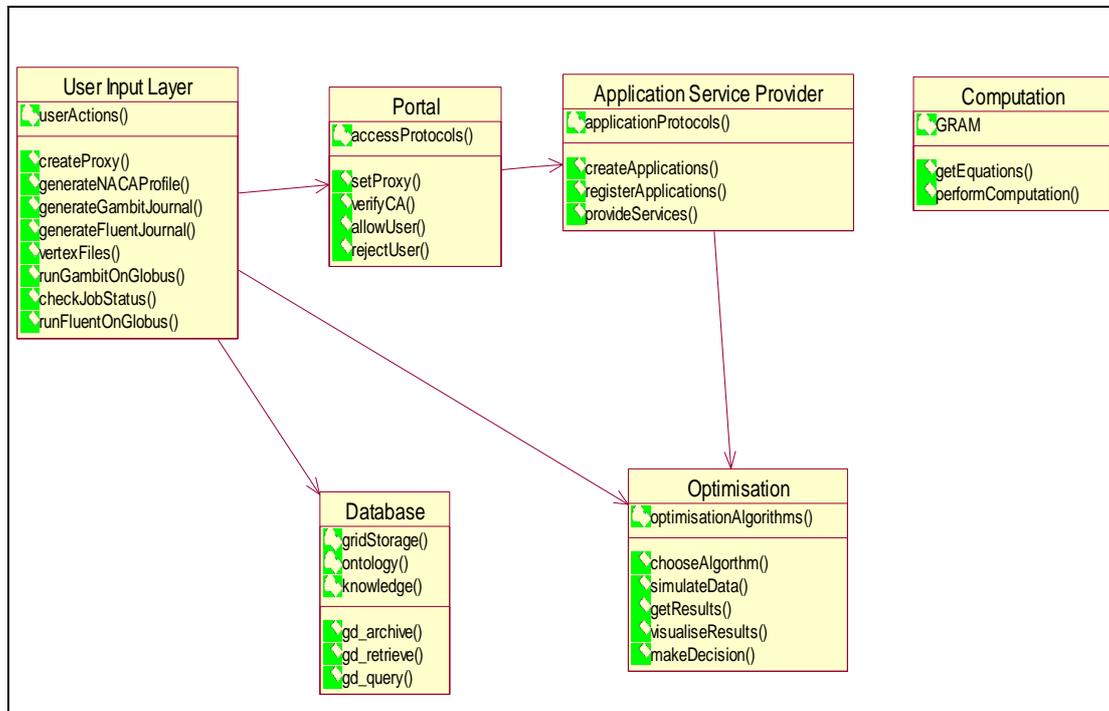


Figure 5.3: Geodise service specifications (Cox et al., 2001)

The main components of the Geodise architecture are portal, computation, optimisation, database, user input layer and application service provider. The portal is the main entry point for users which uses the Globus proxy system of authentication. The computation uses Globus resource allocation manager (GRAM) to submit jobs for computations getting mathematical models from the users. The database stores the data generated from optimisation performed at the optimisation section. Users make

inputs into the system using the input layer and application service layer hosts applications such as Matlab for visualisation of design results.

5.3 MODO services design

Now that some related PSE service specifications have been discussed and their major features understood, this section will provide service specifications for this research. The knowledge acquired in literature on MODO challenges (see sections 2.5.2 and 2.5.4) and MODO design concept (see section 5.1) are used to design these services. One observation in this research is that most PSEs for MODO do not have a service specification and functionality for allowing optimisation experts to build a mathematical model interactively, though they have capabilities to feed in models for optimisation. This research intends to describe how mathematical models can be built interactively and then used to run the optimisation using appropriate algorithm. As mentioned earlier, there are four services that DECGrid will provide for MODO applications. They are mathematical model building service, design parameters input/optimisation service, collaboration service and computational service. These services are proposed based on the requirements for MODO as described in section 5.1. For example, the mathematical model building service guides the designer to enter criteria, design parameters and constraints using graphical user interfaces to come up with a model. MODO experts need to collaborate and share data and make changes. The collaboration enables them to do this. The computational service enables users to use different distributed nodes to submit jobs for computation thus enhancing computational resource utilisation. The design of these services and the functionalities they will provide to the users will be described in the following sections.

5.3.1 Design of mathematical model building service

The first part of service specification is the design of the mathematical model building service. The idea behind building a mathematical model is to inform optimisation experts working on different parts of a complex model on the motivation that links these components to the over all picture of the mathematical procedures so that it is clear what the procedures lead to (Castillo *et al.*, 2002). The mathematical model service consists of quantitative and qualitative interfaces. These interfaces increase the usability of the system as designers have a user friendly GUI to enter design

parameters. The mathematical model builder guides the user to enter design variables that will be used to build the model. Figure 5.4 is the use case diagram for mathematical model creation. Actors include user, expert, Globus Toolkit and client interface.

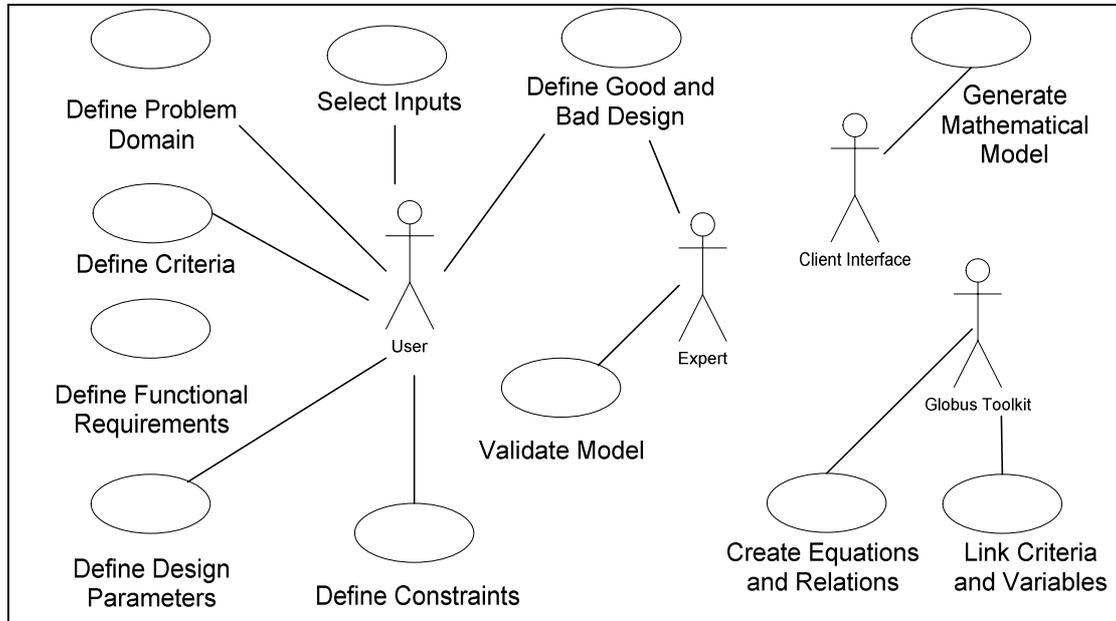


Figure 5.4: Mathematical model building use case diagram

The actions (use case) describe what the prototype is expected to deliver and allow the different users (actors) to perform certain mathematical model building tasks. For example, the user can define design parameters (DPs), constraints, criteria and select inputs. Good and bad design practices can be obtained from the ISO (International Organisation for Standardisation) 9241-11 which defines design usability as the extent to which a product can be used by specified users to achieve specified goals through efficient and effective use of good design standards (Jokela *et al.*, 2002). This use case diagram is used to provide a mathematical model building specification document as shown in Figure 5.5. This specification is the document provided for the programmer. It is the normal specification document as outlined in Globus, FIPER, SORCER and Geodise. An improvement over this will be described in section 5.5 where service specifications will include specifications for high level users of MODO services, not just for programmers. This is to provide emphasise on the issue of grid services as utility.

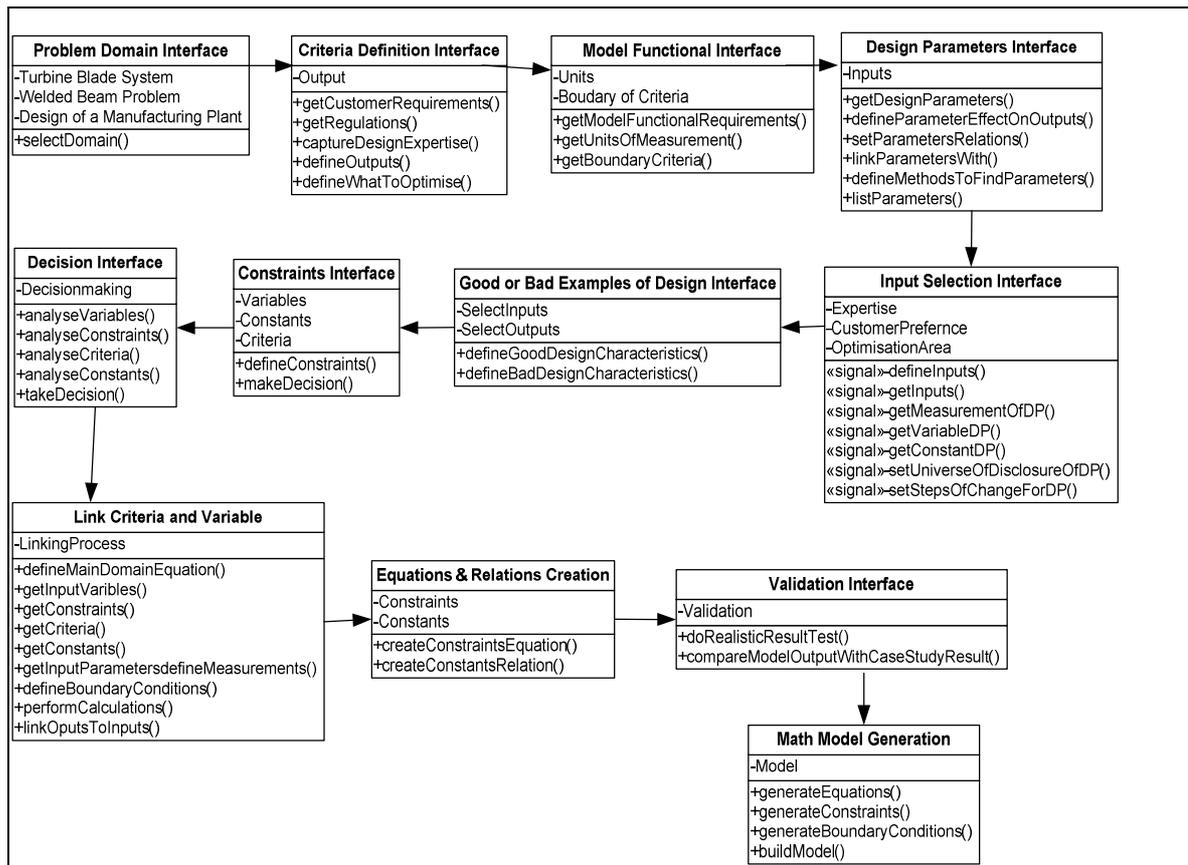


Figure 5.5: Specifications of quantitative mathematical model building service

The same quantitative mathematical model building specification document described in Figure 5.5 as a class diagram is described together with the different actions required at each stage in a sequence diagram. Figures 5.6 and 5.7 show the sequence diagrams that this research used to implement the mathematical model building service. Because the process of the sequence diagram is too large, the diagram is divided into part 1 (Figure 5.6) and part 2 (Figure 5.7).

In describing Figures 5.6 and 5.7, it is important to mention that in coming up with the sequence for building the mathematical model, information obtained from literature was used (Cross and Moscardini, 1995; Castillo *et al.*, 2002). The first stage is the main domain definition. A domain is an area of specialisation that the problem falls under. For example, a problem that is to perform a finite element analysis (FEA) means that the user should select FEA domain and that of cost modelling means cost modelling domain. In this research, the main domain areas are the three case studies. For a given domain, the procedure to build the model is the same. The next step is to

define the criteria. The criteria specify what to optimise and the expected outputs of the optimisation. In the second case study of this research-the welded beam problem, the criteria are to minimise the cost of fabrication and to maximise the end deflection (Deb, 2001). In real practice, customer requirements and designer expertise play a great role in coming up with design criteria (Chen and Lee, 2009). The criteria will determine if the problem requires a quantitative approach or a qualitative one or both. In the design of manufacturing plant layout/floor planning case study, quantitative and qualitative objectives were used. The design parameters which are the inputs are defined using appropriate units of measurements and by applying the boundary conditions that have been identified at the criteria definition stage. The selection of inputs is obtained from a template which defines variables, constants, universe of discourse and design parameters.

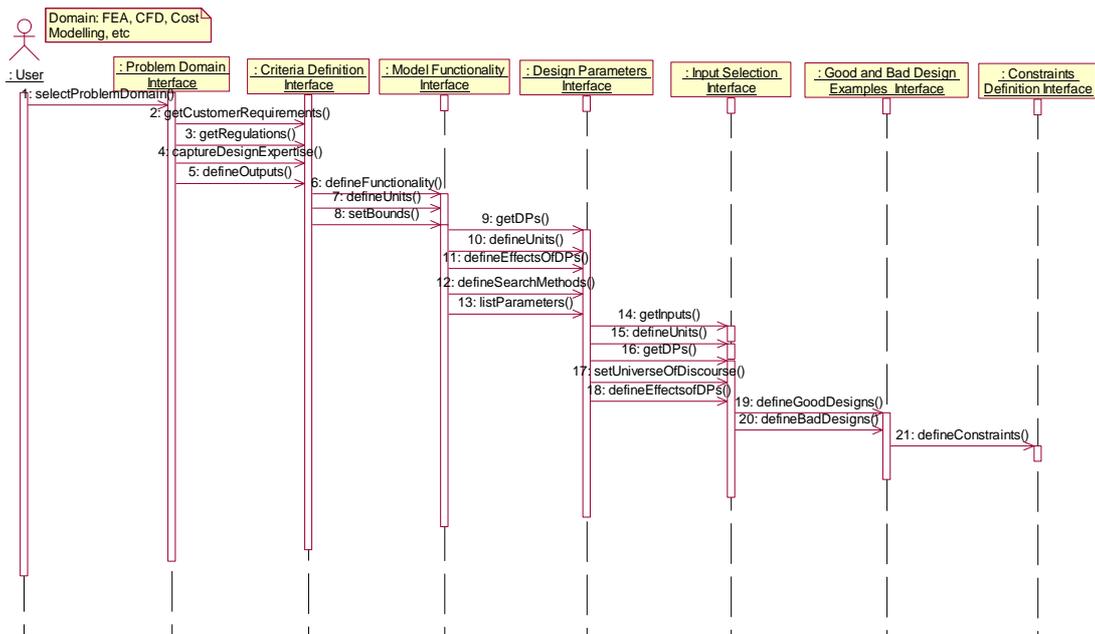


Figure 5.6: Mathematical model building sequence diagram (Part 1)

The constraints are then defined based on the selected variables, constants and criteria. A decision interface is provided at this stage to either continue or repeat the process if the defined variables, constants and criteria are not sufficient for constraints definition (Sundaresan *et al.*, 1993). The system then links the criteria (outputs) and variables. This linking creates the equations and relations for constraints and objectives. The model is validated through sensitivity analysis and realistic results

obtained using case studies in literature. The mathematical model is finally obtained which consists of equations, constraints and boundary conditions.

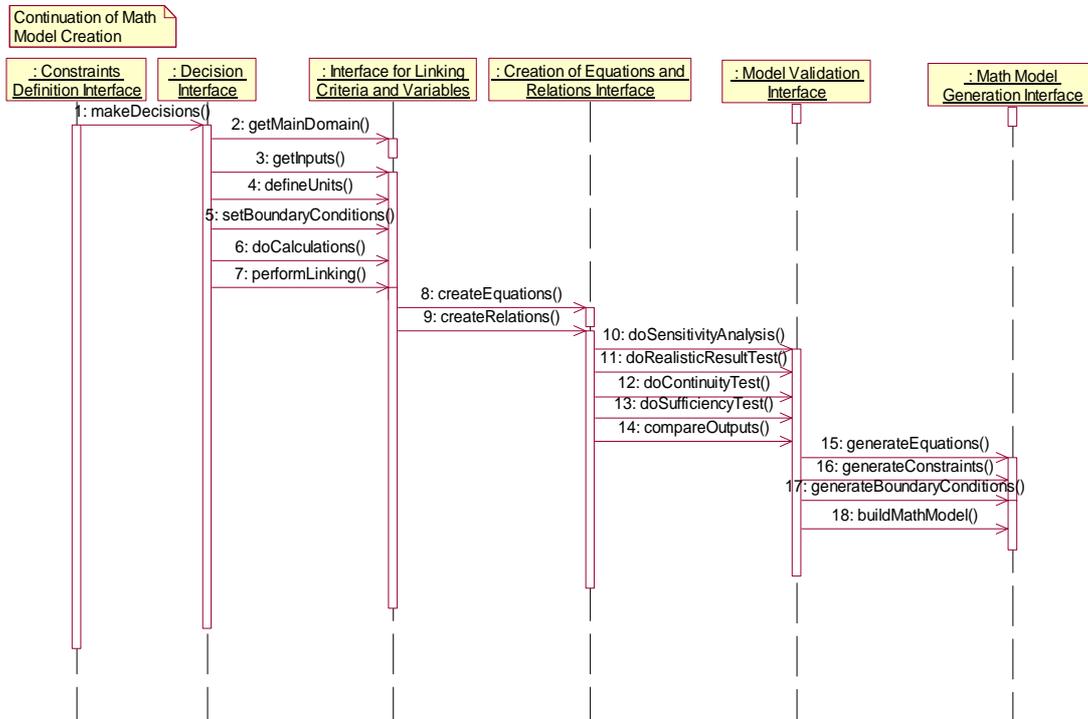


Figure 5.7: Mathematical model building sequence diagram (Part 2)

There were steps that were followed to create the use case and sequence diagrams (Figures 5.4 to 5.7) for the mathematical model building process. These steps were obtained from experts in the academia and industry. The steps consists of (1) capturing the process of mathematical model development (2) capturing the users' requirements that make the tool helpful in mathematical model building and (3) validation of the diagrams. The validation of the diagrams is done as the system is being implemented and tested with users that participated in the validation process.

5.3.2 Design of qualitative mathematical model building service

The descriptions in Figures 5.4 and 5.5 are related to quantitative (Q^T) model building process. To deploy the qualitative (Q^L) model, a specification document is also provided. Figure 5.8 shows the qualitative mathematical model building use case diagram. This shows the Globus toolkit, Qt designer, GNUPLOT, NSGA-II and optimisation engineer as actors. The optimisation engineer as in the Q^T case defines

the domain first. That is, in this case it is the optimisation of design of a manufacturing plant layout/floor planning. This is a generic description of qualitative service. Certain specific steps could be added to deliver different services for different problems.

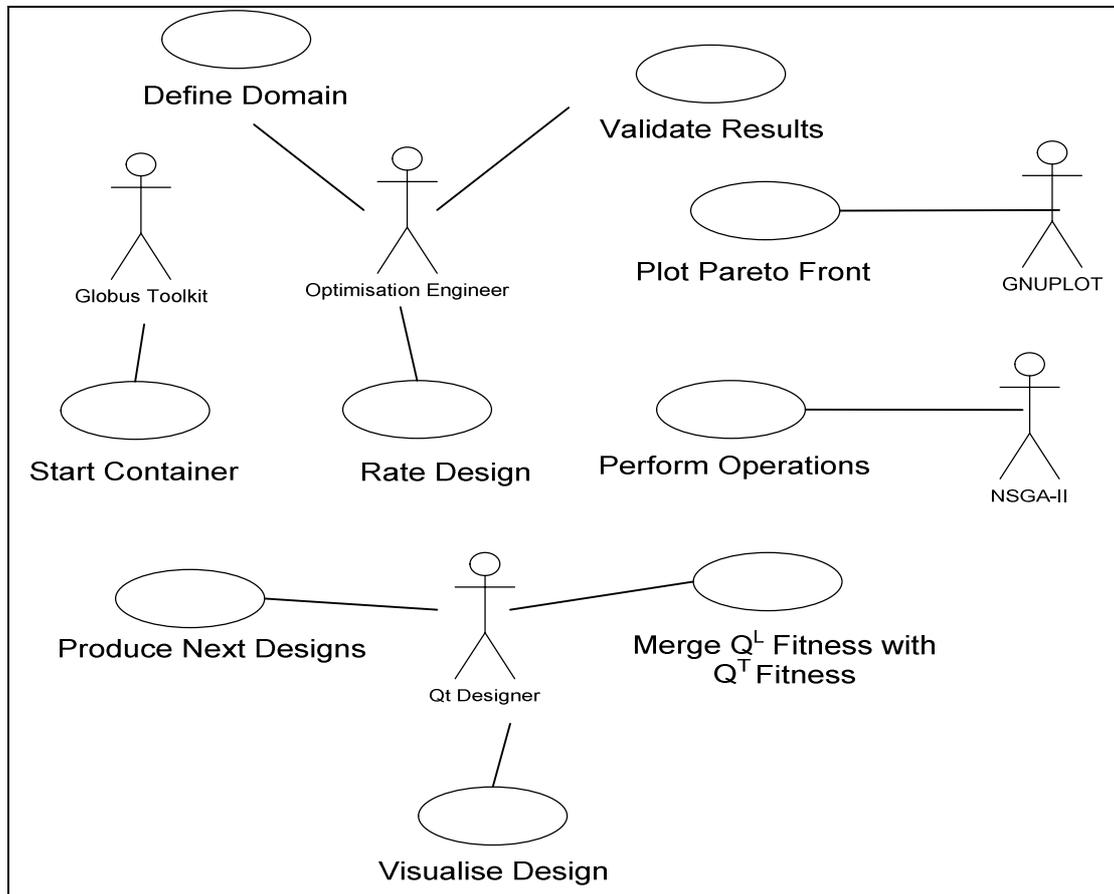


Figure 5.8: Qualitative model use case diagram

In this research, only the third case study, design of a manufacturing plant layout/floor panning will use the qualitative model service. In this case, one node of DECGrid will be dedicated to run the qualitative service and another node will run the quantitative model service. The master node will be used to run the algorithm as well as merge the two services and run the optimisation. Qt Designer is a visualisation package that is used to produce designs for every generation based on the Q^T and Q^L fitness values. The optimisation engineer rates each design produced on a scale of 0 to 9. These values represent the qualitative fitness values. For Qt Designer and any other package used to run, the Globus container must be started. Qt Designer produces designs equal to the number of the population for each generation. It then merges the Q^L values with

the computed Q^T values. After each generation, the NSGA-II performs selection, crossover, mutation and ranking operations on the merged Q^T and Q^L values. The GNUPLOT (the software package that comes with NSGA-II to plot graphs) plots the Pareto front. The optimisation engineer then validates the results. Figure 5.9 is a description of the qualitative model service process as a sequence diagram. For the design of the manufacturing plant layout case study, the service allows the distributed users to run the optimisation problem and view design for certain generations. In the rating, 0 means VERY BAD design and 9 means EXCELLENT design. The practical application and results will be discussed in detail in chapters 7 and 8.

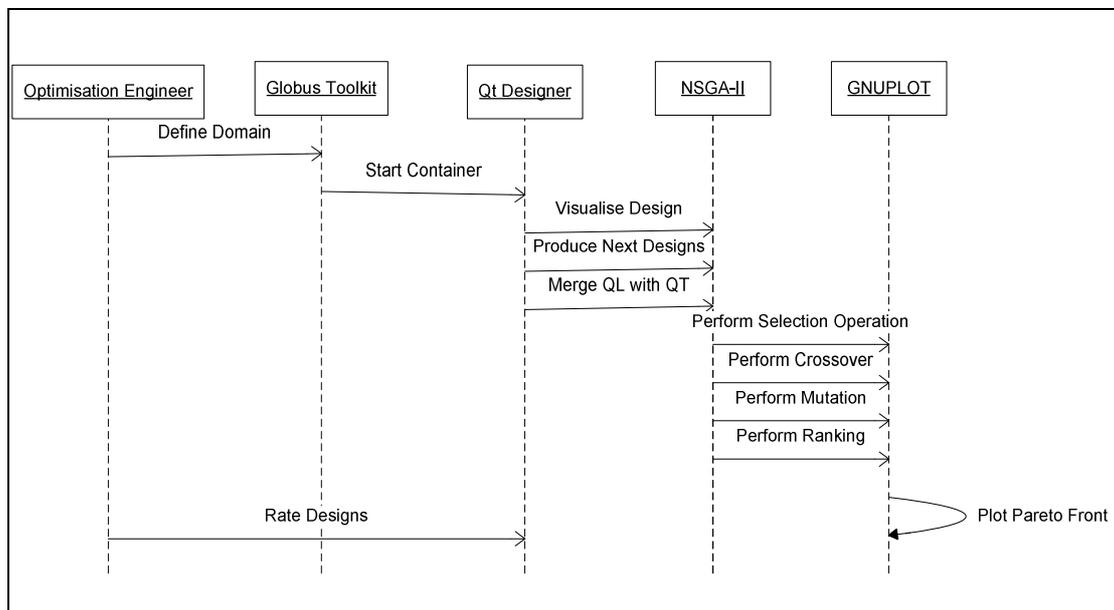


Figure 5.9: Qualitative service specification sequence diagram

Qt Designer is used to display the different designs created in each generation. Qt Designer is an open source software that has visualisation capabilities and the codes are written in C++. DECGrid uses Globus middleware and Apache web server to execute services that use the rating values entered through the Qt Designer interface. Figure 5.9 is a generalised representation of qualitative design interface. This research used the ratings (qualitative fitness values) and quantitative values to obtain the Pareto front for the third case study.

The steps taken to obtain Figures 5.8 and 5.9 are (1) definition and understanding of the qualitative problem (2) identification of visualisation interface for designs and (3)

identification of input (rating) interface for users. The validation is done by experts who participated in the validation of the prototype. It was also validated by discussing the results and workability of the system with the original source of the problem (Brintrup, 2007).

5.3.3 Design of collaboration service specification

DECGrid is designed to serve distributed users like most grid PSEs. Collaboration among distributed experts is one feature that will be demonstrated during validation of the prototype. Collaboration service is the term used for any collaborative platform that enables users to accomplish specific goals within a virtual organisation. This research provides a collaboration service that allows design engineers to share data and make suggestions on how to improve the design. Changes can be made on parameters before a final decision is taken by the experts. The collaboration consists of distributed database repository through which resource owners grant access rights to collaborating partners. The scheduling and querying of data is managed using Condor scheduling system and PostgreSQL database server (an open source database server) query capability. The service gets started when the Globus container and web server are started. In this research, the collaboration is used for computational steering in the case studies. Grid PSEs use collaboration to enhance efficient visualisation in collaboration and computational steering (Liu *et al.*, 2005). Figures 5.10 and 5.11 describe the collaboration service specification use case and sequence diagrams respectively.

In Figure 5.11, the relationship between resource owner and resource user is described in the collaboration. Resources are first registered by the resource owner in the Globus Information Index Service (GIIS) and are made visible through the Web Monitoring and Discovery Service (WebMDS). The resource owner creates access to users using the grid map-file in the grid security. Through GridFTP, data can be transferred between collaborative users. Efficient scheduling of the shared resources is handled by the GRAM and Condor which is usually referred to as Condor-G resource management system. The building of the mathematical model described in section 5.3.1 is made possible through the collaboration service.

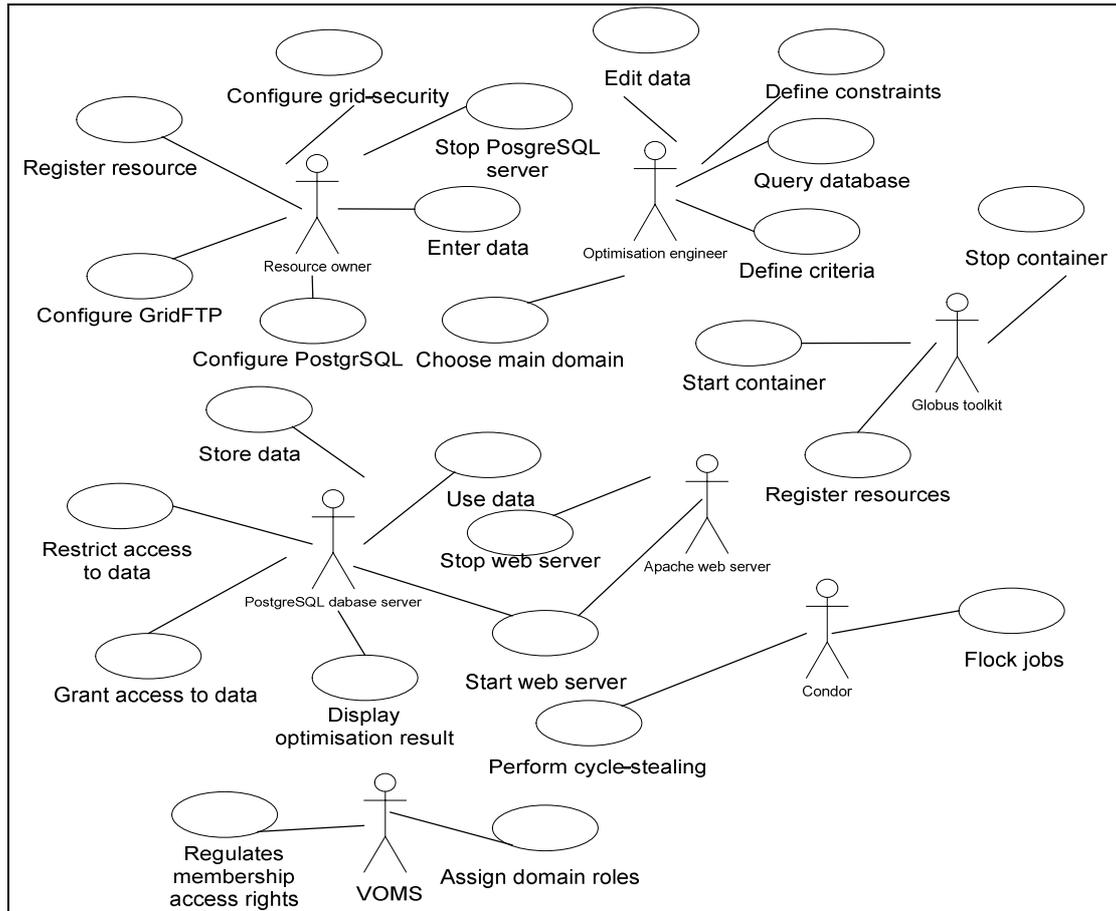


Figure 5.10: Collaboration service specification use case diagram

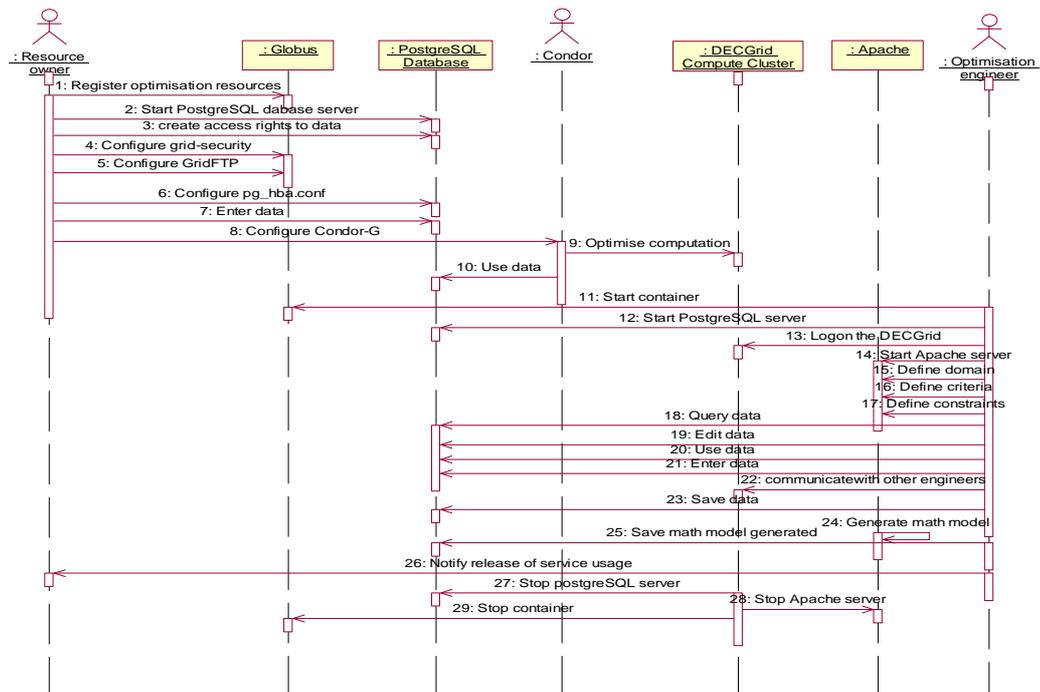


Figure 5.11: Collaborative service specification sequence diagram

The steps used to obtain Figures 5.10 and 5.11 are (1) identification of service provider and service users' requirements (2) identification of Globus, Condor and PostgreSQL capabilities for collaboration and (3) consideration for restriction of access rights for users. The validation is carried out using scenarios where users build a model and submit it as a job. The model is registered as a resource in the Globus Information Service (GIS) and gets published in the WebMDS. Users could view it and optimisation algorithm available.

5.3.4 Design of compute service specification

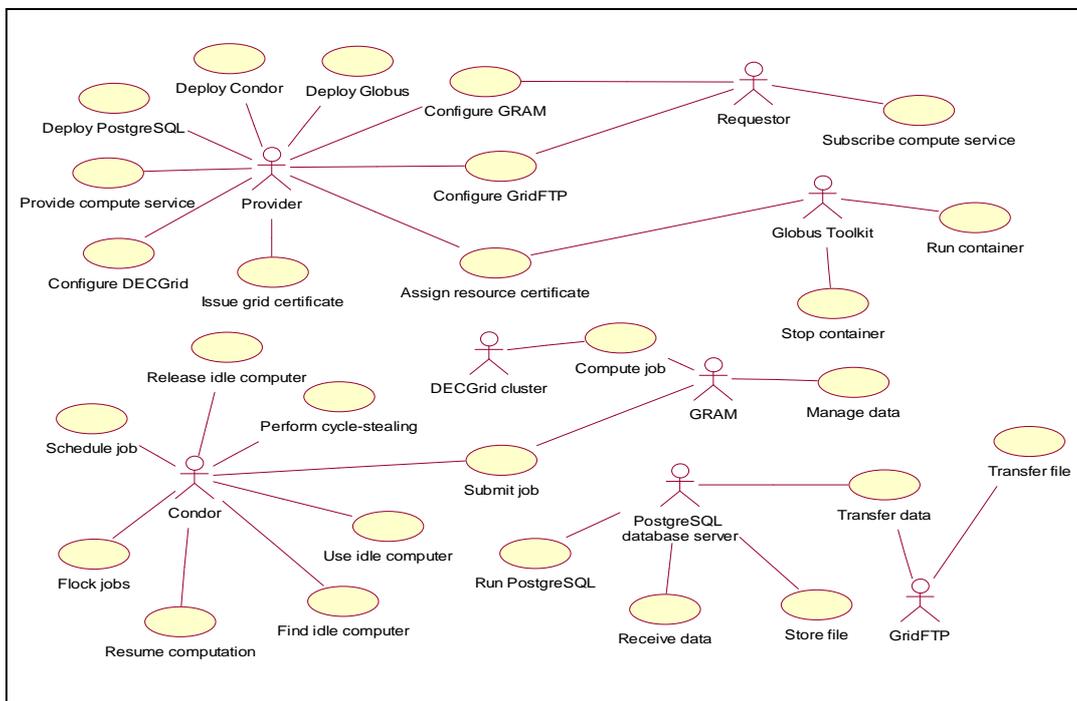


Figure 5.12: Compute service specification use case diagram

One of the services provided by this research is the compute service. Computational power is the traditional and first motive behind the grid concept (Foster and Kesselman, 1999). This service uses Condor scheduler to flock optimisation jobs to idle nodes. This ensures computational throughput and speeds up the optimisation process. Figures 5.12 and 5.13 are the compute service specification use case and sequence diagrams.

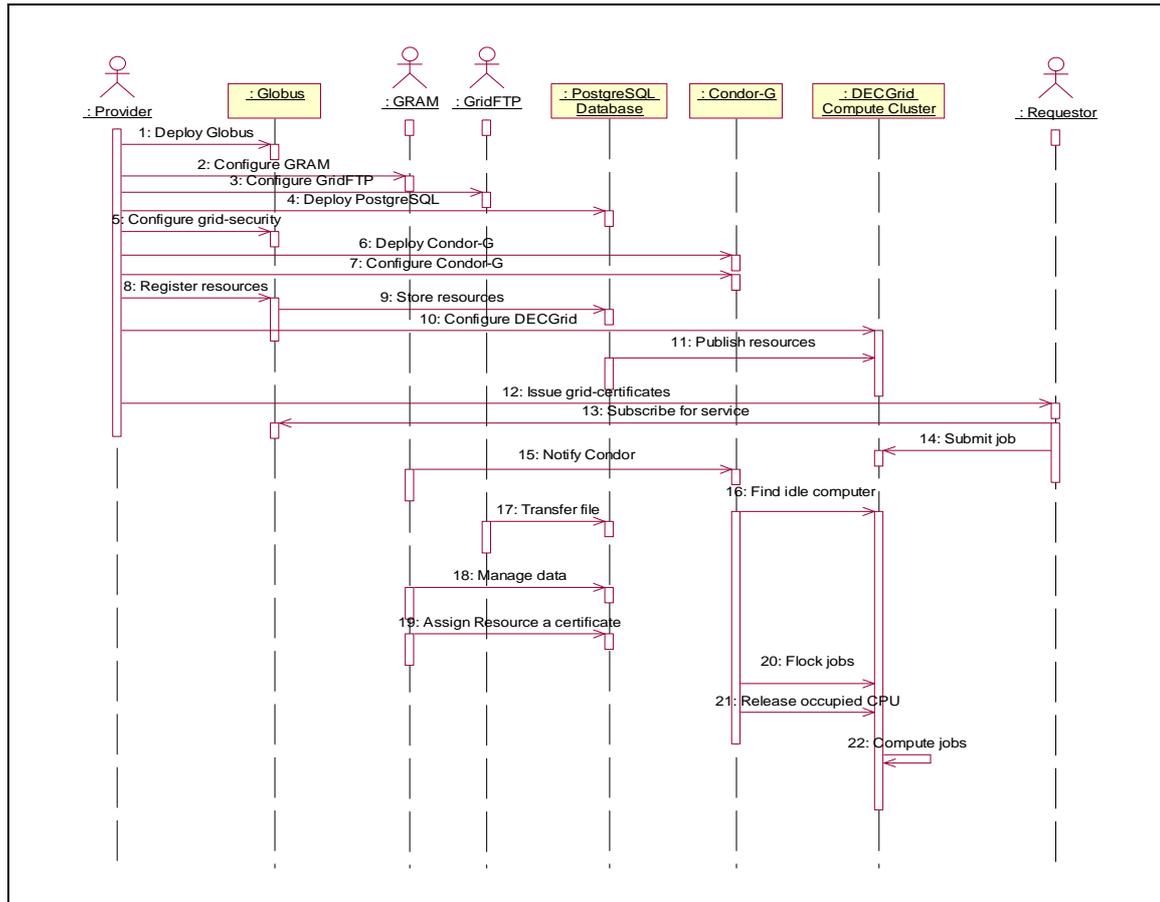


Figure 5.13: Compute service specification sequence diagram

The main actors in the use case diagram are provider, requestor, Condor, DECGrid cluster, Globus toolkit, PostgreSQL database server and GridFTP. The provider deploys Condor (scheduler), GridFTP (for secure transfer of data), GRAM (for resource management) and provides them as computational services for optimisation experts (requestors) to use. The provider also issues certificates of authentication and authorisation to the users. This ensures that the system is secured from intruders and also ensures the integrity and consistency of design data shared among collaborators. PostgreSQL database is used to store design variables so that they can be reused when the need arises.

The concept of scheduling jobs across idle desktops in companies or research organisations is becoming an innovative way of cutting cost of buying and maintaining supercomputers. This can be done by combining the power of platforms running different operating systems and heterogeneous distributed machines to improve efficiency of computation and data access (Chung and Chang, 2009).

5.3.5 Optimisation parameter input service

Sections 5.31 to 5.3.4 described the mathematical model building, collaboration and computation services. These section discusses the process of carrying out optimisation using the facility described. The main actors in the optimisation and parameter input service are optimisation engineer, Globus toolkit, Apache web server, explicit/implicit interfaces and NSGA-II parameter interface. Figure 5.14 is the use case diagram for the optimisation parameter input service.

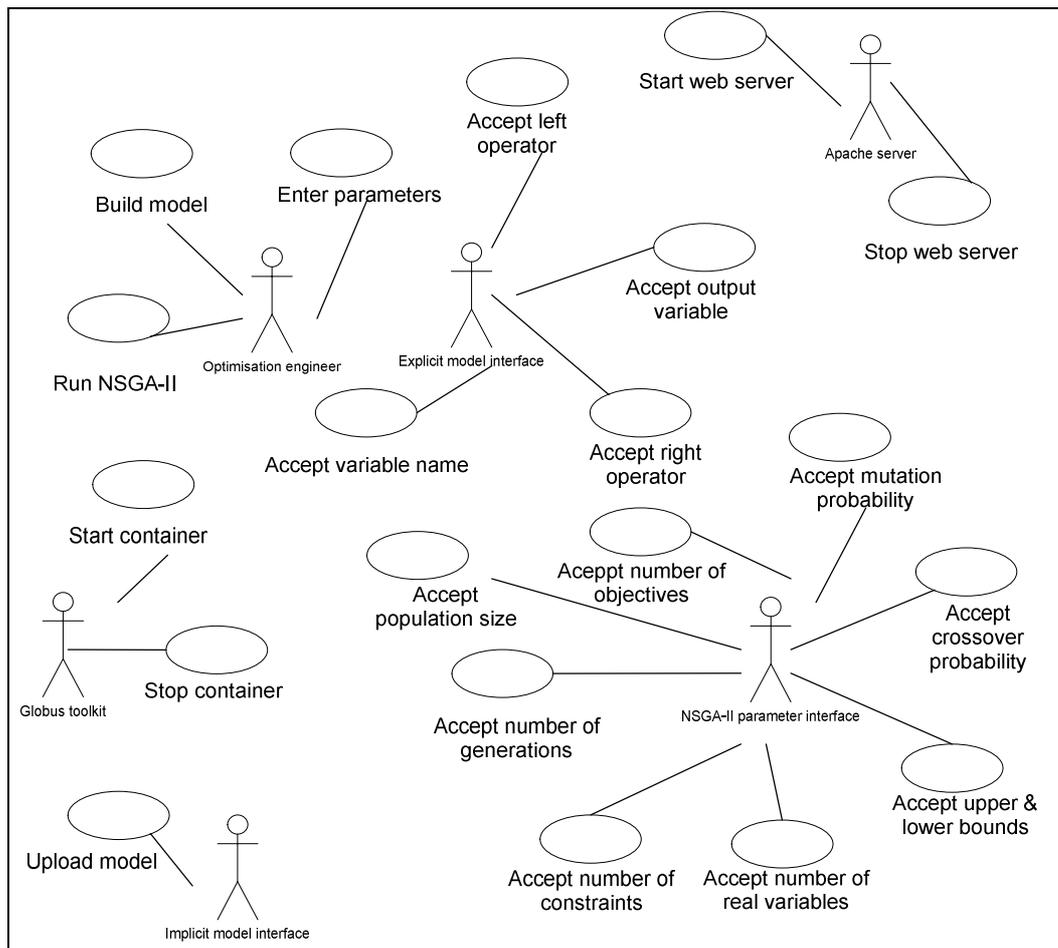


Figure 5.14: Optimisation parameter input use case diagram

The Globus container (the name for the system that activates Globus) is started together with web server for both grid services and web services to run. The optimisation engineer creates explicit mathematical model using variable names, mathematical operators and symbols. Implicit models can be uploaded from files. The NSGA-II parameter input interface accepts parameters such as population size, number of generations, number of objectives and number of constraints from users.

Figure 5.15 describes the details of actions performed by actors as a sequence diagram. Figures 5.14 and 5.15 are obtained by studying the steps used to run optimisation using NSGA-II user manual. The process is validated by an expert from the research centre that developed NSGA-II. The user interfaces for Figures 5.14 and 5.15 are described in section 5.4. The idea is to make the system useful to designers. Normally users need to issue Linux commands from command prompt.



Figure 5.15: Optimisation parameter input sequence diagram

5.3.6 Resource aggregation

The grid also has capabilities to aggregate computational resources to offer them as services. Recently this concept of aggregating computational resources such as computational power and data storage and providing them as utility to users is known as cloud computing (Buyya *et al.*, 2008). In this research, the concept of aggregating computational resources is done by making the resources on the nodes of DECGrid (processors, disk storage, algorithms, mathematical model, etc) transparent and accessible to all users at different nodes to ensure efficient and optimum utilisation of

limited optimisation resources. From the concept of resource aggregation, this research defines aggregation in terms of MODO resources as:

MODO Aggregation (MODOA) is the pool of collection of uniquely dynamic multi-objective optimisation resources and services contributed by participating members of a virtual organisation and are made transparent to distributed design experts through a secure grid portal for collaboration.

Figure 5.14 describes the MODOA sequence diagram. To register a resource, a configuration file is created in XML and is saved in the \$GLOBUS_LOCATION/etc/globus_wsrf_mds_aggregator/ location. The command mds-servicegroup-add is issued to add the resource. The Globus WebMDS Index Service is used to publish the aggregated resources and services for use by the virtual organisation. It is assumed that these resources are contributed by the distributed grid sites for optimisation purpose. Service registrations to the aggregator framework are configured by using XML based configuration file and the registered resources available. Query and subscription actions are configured so that users can query and subscribe for resources from the MODOA.

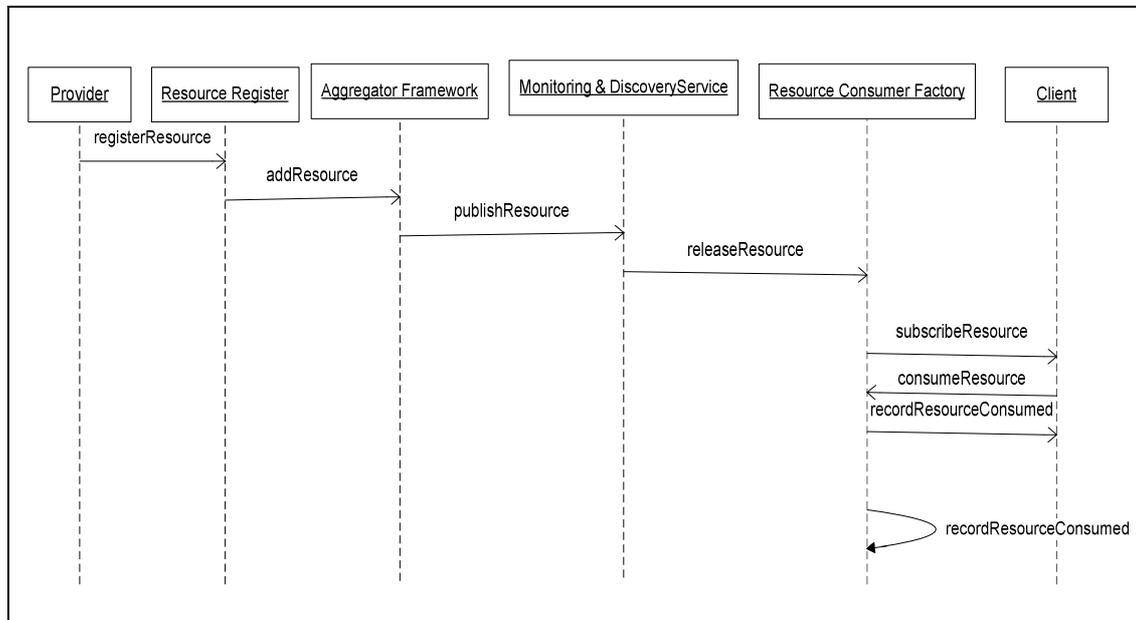


Figure 5.16: MODO resource aggregation sequence diagram

The Globus resource consumer factory allows users get access to the resources. In this way optimisation resources are published by providers and used by requestors.

The steps used to accomplish Figure 5.14 are by following the steps of registering and aggregating resources in Globus Toolkit (www.globus.org). The steps involve a provider registering a resource, a user viewing the resource, subscribing the resource and using it. The validation is done during deployment with users who participated in the validation. A factory pattern resource management allows MODO resources to be uniquely registered by providers in the MODO service factory and the resources get published at the MODO WebMDS index service where users can view and subscribe for the resources. Factory pattern design methodology has the advantage of putting resources separate from the operations and the results of the operations for speedy process and access to aggregated resources (Lynden *et al.*, 2009). A MODO subscription interface allows users to invoke the notification service. Grid is supposed to be a resource sharing infrastructure for research and business. The notion of resource aggregation is exploited in this research to ensure resource sharing for scientific and business objectives among designers. To understand the requirements for grid services that will suit MODO applications, the Globus documentation on the configuration files for aggregator source and Globus information service were studied. This enabled the researcher to customise the files used for registering resources and making them available through the WebMDS.

Section 5.3 and the subsections under it described the design that helped the research to provide MODO capabilities on the grid. This accomplished the objective of understanding grid requirements to provide specifications for designers to support MODO. Section 5.4 described the user interfaces as implemented in chapter 7.

5.4 MODO interface design

DECGrid has interfaces for optimisation users to interactively work and create the models, enter design parameters and collaborate. The usual optimisation carried out in Linux grid environment is done by issuing commands and inputs through the command prompt environment which may not be comfortable and efficient for engineers in design optimisation. Graphical user interfaces (GUI) are designed in this section and will be implemented in chapter 7. The GUI provides a web and grid-enabled interactive and user friendly environment for MODO experts. The different interfaces will be described below.

5.4.1 Definition of MODO service interface for domain disciplines

The three case studies that are used to validate the results of the research are described as domain disciplines. The first interface in the prototype is for the user to choose a domain area in which he or she wants to perform optimisation. This is important as different algorithms are well suited for different domains. Though this research is only using NSGA-II for the three case studies, it is good that a generic interface is provided in case another algorithm other than NSGA-II is to be used in the future. This makes the system more robust and scalable. FIPER also has GUI that makes it attractive to users and has capabilities that encourage the creation and use of web and grid-enabled models for different disciplines (Lee *et al.*, 2009).

5.4.2 Definition of MODO service interface for users

The DECGrid prototype provides GUI using HTML (Hypertext Mark-up Language), JavaScript and PHP programming languages. C is used for the NSGA-II optimisation algorithms to write codes that perform the actions that the models built are expected to deliver. There are two interfaces for getting models into the NSGA-II. One interface helps design experts to build explicit mathematical models while the other allows complex implicit mathematical models to be uploaded from a file unto the NSGA-II. The first interface which is the explicit model building interface will be discussed first. After choosing a domain, to build an explicit model for the domain, four generic fields are used. These are variable name (independent variable or constant), left operator, right operator and output variable (dependent variable). The left and right operators are mathematical operators such as plus sign, minus sign, brackets and multiplication signs. To build a model, a user will enter variable and/or left/right operators and submit continuously until all input variables are complete for an equation/inequality. The last action to generate an equation in a model is to enter the output variable together with equality sign (=) and click submit. This will create an equation. This process is repeated for the next equation until all equations and relations in the model are created. This process can be done by different experts on different parts of the model and can still be merged to obtain a single model for a problem. Figure 5.16 below shows the design of the explicit model building interface.

Explicit Model Building Interface

Variable name:

Left operator(s):

Right operator(s):

Output variable and assignment sign:

Figure 5.17: Explicit model generation

**NSGA-II Input Parameters
Main Parameter Interface**

Enter population size:

Enter no. of generations:

Enter no. of objectives:

Enter no. of constraints:

Enter no. of variables:

Figure 5.18: NSGA-II input parameters (R_Var=Real Variable) (phase 1)

The system is designed in a way that allows the optimisation engineer to view the model built by clicking the Display button. When the engineer is satisfied with the model, he/she can link the model to the problemdef.c file (this is the file NSGA-II uses to pick models and run optimisation) in NSGA-II code by clicking the Append button. NSGA-II requires a set of parameters to run the optimisation of any model.

Because of this, grid-enabled input parameter interfaces have been designed to accommodate this feature. These interfaces make life easier for the designer than using the conventional command prompt input interface. This interface (Figure 5.17) is linked to optimisation processes where a parameter can be changed to demonstrate how sensitive or otherwise it can be to the results. To perform the optimisation for the model obtained, the NSGA-II Optimisation button is clicked to first obtain optimisation input parameters. When this button is clicked, the first input parameter interface (Figure 5.18) appears.

All evolutionary algorithms (EAs) need population size, number of generations, number of objectives, number of constraints and number of variables that constitute the model and constraints. For example, EAs need to know the population size for the selection operation in which good solutions are duplicated and bad ones eliminated keeping the population size constant (Deb, 2001). In addition too small and too large population sizes may result in misleading results and so NSGA-II needs a moderate population size to produce reliable solutions. Thus the population size is proportional to the complexity of the model built (Deb, 2001). In this research, explicit models are relatively simple models while implicit models are complex models. The distributed experts working on models make decision on the values of each parameter before entering them. Figure 5.17 is the first phase of the input parameters. Information on the population size and the number of generations, objectives, constraints and variables is obtained. The system saves the information to PostgreSQL database as well as to the parameter input text file which is redirected and used to run the NSGA-II. The Generate R_Var (Generate Real Variable) command button uses the value of the last input variable (number of variables) to generate the upper and lower bounds for each variable. For example, if there are n numbers of variables, the lower and upper bounds of each of the n variables will be entered and saved immediately below the first five parameters obtained from Figure 5.17 in the same input parameters text file as in Figure 5.18. The design of the interface which allows the design expert to enter lower/upper bounds for n real number of variables is shown in Figure 5.19.

Real Variable Boundary Values

Enter lower bound of variable 1:

Enter upper bound of variable 1:

⋮

Enter lower bound of variable n:

Enter upper bound of variable n:

Figure 5.19: Boundary values for real variables

The next input interface is shown in Figure 5.20. This takes the service to phase 2 of the input parameters procedure. Figure 5.19 is the interface for this phase. This interface allows the user to enter probability of crossover of real variable, probability of mutation of real variable, distribution index of crossover variable and number of binary variables. The inputs are appended to the same parameter file containing the previous parameters and also get saved in the PostgreSQL database for reuse. The Generate B_Var (generate binary variables) button takes the last value of the parameter in Figure 5.20 (number of binary variables) to generate the number of bits, minimum number of binary variable and maximum number of binary variable for each binary variable. Figure 5.21 shows the design for this interface.

NSGA-II Input Parameters Phase 2

Enter probability of crossover of real variable:

Enter probability of mutation of real variable:

Enter distribution index crossover variable:

Enter no. of binary variables:

Figure 5.20: NSGA-II input parameters (phase 2)

Properties of Binary Variables

Enter no. of bits for binary variable 1:

Enter minimum no. of binary variable 1:

|

Enter minimum no. of binary variable n:

Enter maximum no. of binary variable n:

Figure 5.21: Properties of binary variables

The properties for the binary variables are also appended to the input parameters text file. In this research, binary variables are not used; only real variables are applicable for the three case studies. But the functionality for binary variables is implemented so that a generic system is built to cater for different category of users. The last phase (phase 3) of the input parameters interface is shown in Figure 5.22.

**NSGA-II Input Parameters
Phase 3**

Enter probability of crossover for binary variable:

Enter probability of mutation for binary variable:

Enter choice to display real-time data gnuplot (1 or 2):

Enter the 3 indices (obj1, obj2, obj3) for x, y, z axes:

Enter polar and azimuth angles (angle1, angle2):

Figure 5.22: NSGA-II input parameters (phase 3)

The concluding NSGA-II input parameters in Figure 5.22 are probability of crossover of binary variable, probability of mutation of binary variable, display choice (graphical or only text results), indices of the objectives and polar/azimuth angles. These parameters are also saved (appended) again in the input parameters text file as the previous parameters.

The input parameters are ready for NSGA-II to perform the reproduction, crossover and mutation operations. At the end of the last generation, the GNUPLOT capability in NSGA-II plots the Pareto front of non-dominated solutions. The interfaces described make up the parameter and NSGA-II optimisation service.

Figure 5.23 is the interface that uploads mathematical model for implicit models. In this case, the model is obtained from a file and linked to NSGA-II problemdef.c file. The case study for the turbine blade cooling system is an implicit model and this interface is used to upload the model for optimisation. The remaining two case studies (welded beam problem and design of a manufacturing plant layout) are both explicit models and the explicit interface is used for building the models.

Implicit Model Interface

Enter model file name:

Figure 5.23: Implicit model generation

The model is uploaded and the same procedure for obtaining parameters as in the case of explicit model and is followed. This means that the design ensures that the system implements the same actions as is described in explicit models. Figure 5.24 describes

the high level relationships of the services discussed and the interfaces as a tree diagram.

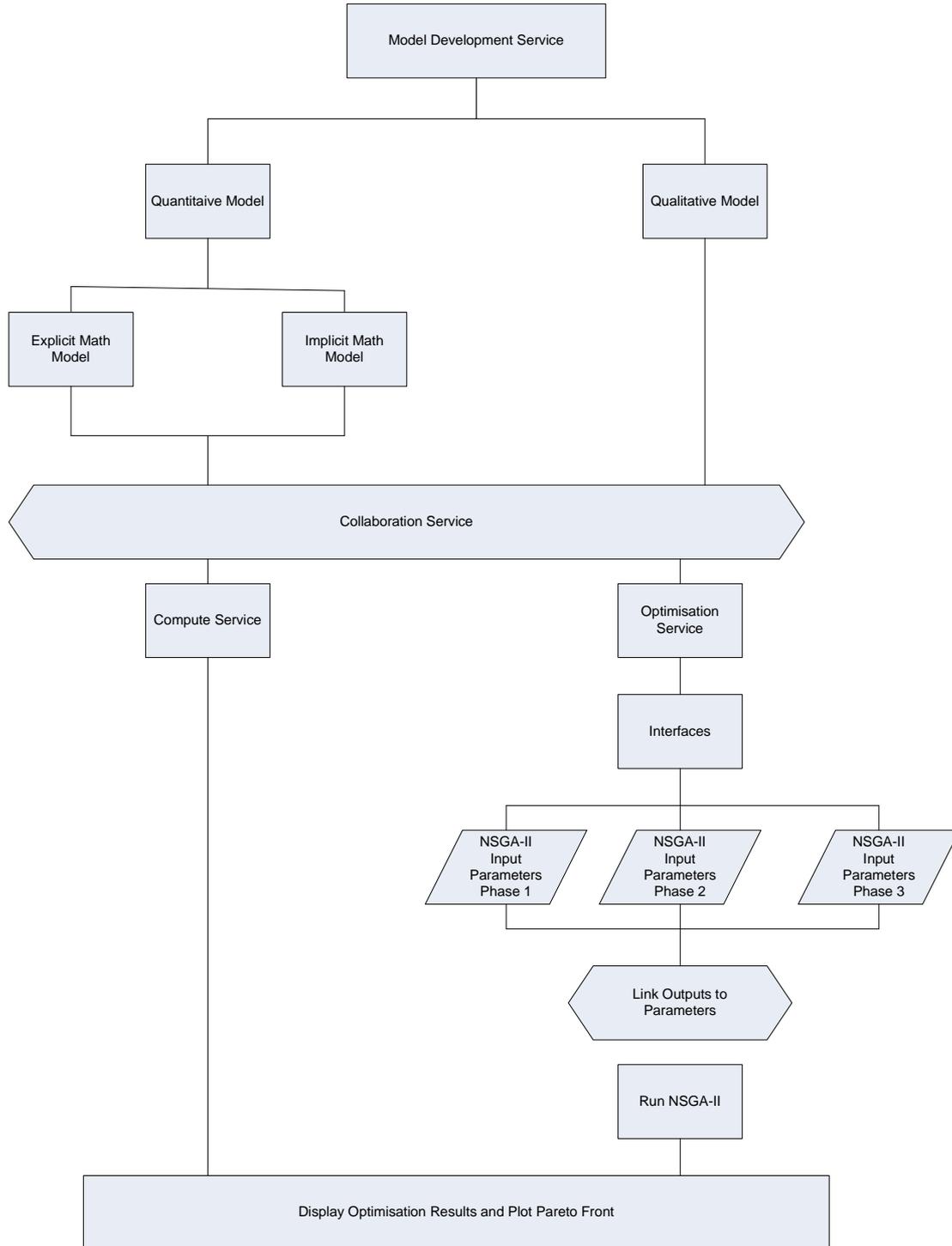


Figure 5.24: Tree diagram of the services and interfaces

5.5 Research contribution in service specification

The design of most grid service specifications concentrate on object-oriented specifications for the programmers. For example as discussed in section 5.2, the

SORCER service specification (Figure 5.2) described only the service functionalities that programmers need to implement. From literature and industry survey, there is no document that shows the binding relationship between the providers and users of the service. Although the Globus toolkit which Geodise used as middleware has implementations for fault tolerance and service time availability, there is also no high level provision for service agreement between service provider and user in the service specification. What is obtainable is the web service description language specification. This research provides a structured service specification for programmers as well as service definition and specifications for end users of the grid services in this case for optimisation engineers. This is in line with the initial concept of the grid as a utility resource sharing infrastructure like the electricity grid.

The MODO service specification document for programmers that is used to develop the prototype is shown in Figure 5.25. This document describes how service providers as organisations offer services to requestors. The document should include a service level agreement which is binding on both parties. This is where reliability and availability of resources are described as part of the incentive to join the grid environment. As discussed already, the MODO aggregator consists of aggregated resources for sharing among distributed users. In addition, the collaborative procedure for building a mathematical model within the MODO service specification and testing its workability using the optimisation service interface gives the research a scientific and academic perspective. This optimisation interface guides the optimisation engineer to carry out optimisation. This means that engineers without optimisation knowledge can conveniently perform optimisation task. This is because the interface starts with the selection of a domain then guides the user on how to enter criteria, design parameters and constraints to run the optimisation. The process of service specification for end users of the services will form part of the proposed MODO framework in chapter 6.

The emphasis is on resource management and quality of services provided. IBM has implemented some form of SLAs in its web services systems and the OGF (Open Grid forum) has also implemented the WS-Agreement in Globus Toolkit but this is a web service description language based on extensible mark-up language implementation. A high level description of such SLAs is required for end users to be

aware of the terms and references before going into any agreement. Major components (classes) of Figure 5.25 are described below and how they relate to the framework in chapter 6.

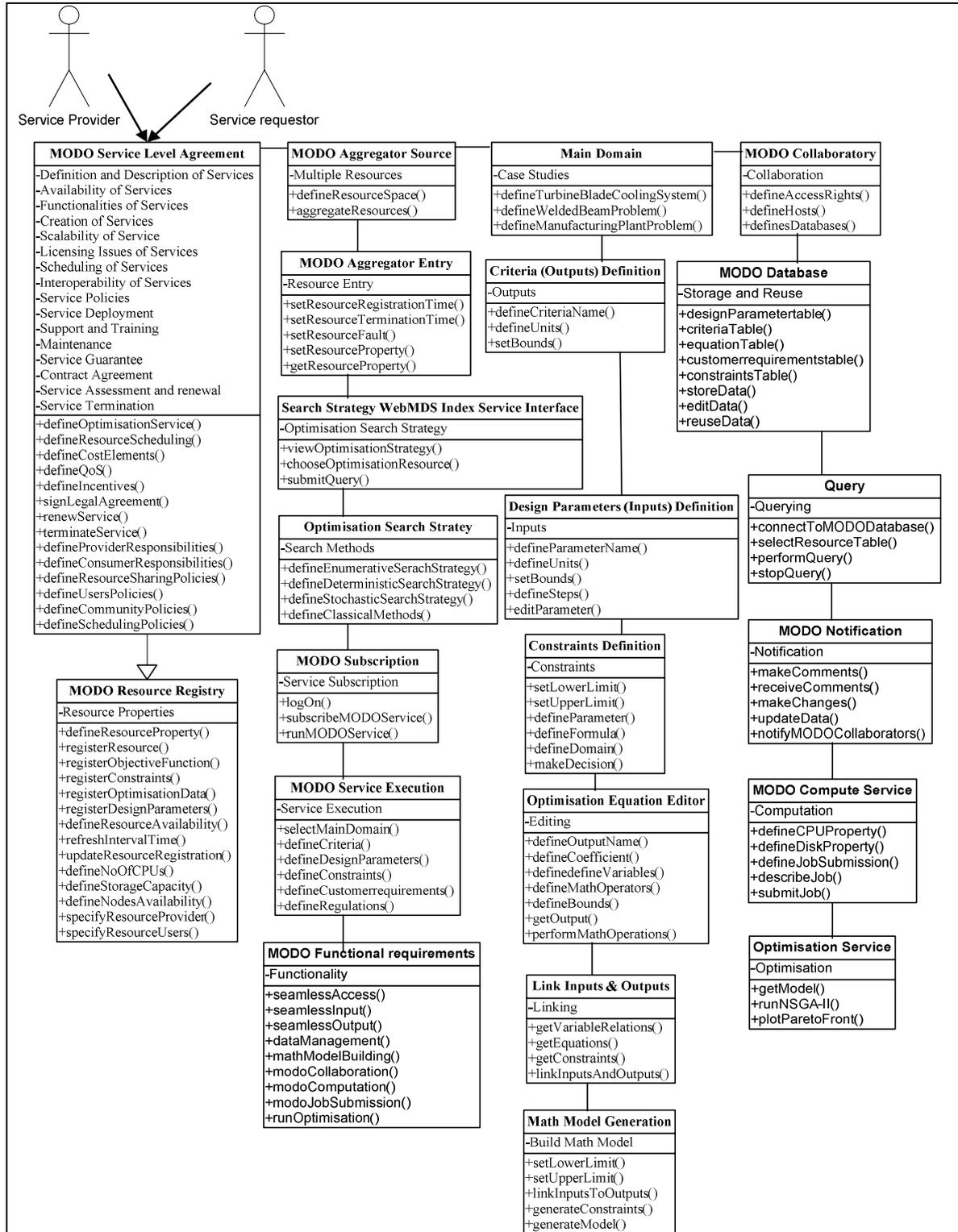


Figure 5.25: MODO service specification

The objective of including service level agreement (SLA) in the service specification documentation is to promote and establish in clear terms the baseline agreements for service functionalities and responsibilities of service providers and requestors in the grid community. These agreements may cover general aspects of grid services or specific areas of services provided. This research provides a generic structure for SLA but with more emphasis on MODO applications which is part of Figure 5.25.

5.5.1 MODO service level agreement

A typical SLA structure may consist of main items under the definition/description of services, creation of the services, interoperability, scalability, scheduling, functionalities, policies, service deployment, support, maintenance and training. Others are guarantee of services, legal contractual agreements, service assessment/renewal and termination of services. Definition and description of services refer to the types of services and how the services are provided and subscribed for including the technologies. It is important for the grid services provided to follow standard protocols that allow interoperability among diverse users running different platforms. Grid projects start small but with the intension of expansion, so scalability and robustness of the grid is important. Policies are important to guide users' access, resource scheduling and resource sharing for the community. This functionality is implemented in chapter 7 using virtual organisation membership service (VOMS) capability. VOMS defines roles and assign access rights to members of grid service users through domain configuration. In service deployment, support and training are important to sustain the services deployed. Grid systems are complex and unstable and therefore support is an important aspect of grid community. In the course of this research, two nodes of DECGrid failed and had to be reconfigured. Majority of advertisements for grid roles in research and companies are support roles. Service guarantee is based on the fault tolerant system in middleware and scheduling system. When the user is confident that there is service guarantee, a contract agreement can be signed. In the contract agreement, clause for service assessment, payments, renewal and termination are included. This aspect of service document takes cognisance of the responsibilities of providers and users and makes the grid a proper utility infrastructure.

During industry survey, the researcher asked some software company officials and users if they are ready to sign such contractual agreements for services. The companies said contractual agreements are based on licensing issues for using a service for a period of time but they do not sign contractual agreements for services on-demand (utility) for now for applications. However, companies provide storage and processing capabilities for users and charged them per usage; a concept earlier described in chapter 2 as cloud computing.

5.5.2 MODO resource registry

Globus resource registry is customised to register MODO resources. The properties of the resources are defined. For example NSGA-II algorithm is registered as a resource. The properties of NSGA-II such as being a genetic algorithm, requires mathematical model and parameters such as population size, number of generations, number of objectives, number of constraints and so on are indicated. The resource and its properties are defined in an excitable extensible mark-up language (XML) file. This resource is displayed in the Web Monitoring and Discovery Service (WebMDS) for users to see and subscribe. To add a resource in the registry, the command `mds-add-resource` is used. Resources can be updated and removed. Resources such as number of processors and built in Globus services such as GRAM (Globus Resource Management) and RFT (Reliable File Transfer) are automatically registered when Globus is deployed and by starting the container, all these resources appear in the WebMDS.

5.5.3 MODO aggregator source

The aggregator source enables all MODO resources registered by different users on distributed nodes to be aggregated and made visible to all users. The sharing of these resources is controlled using individual's grid identity in the grid map-file of each node. If a user's identity is omitted in the map-file of a node, it means that user cannot use the resources of that node. The aggregator entry is used to set when resources are registered and when they will be terminated. Report for faults is also configured here. This ensures that service MODO experts know before hand the availability of resources and check if this conforms to the terms agreed upon in SLA. Users however can increase the time they want to use a resource or terminate the resource usage.

5.5.4 MODO search strategy interface

This is the interface that is used by MODO experts to search for resources based on their properties. This interface is customised from the Open Grid Services Architecture Browser graphical user interface. The interface has fields to search for resources using their service data elements or grid service handle. For example to search for genetic algorithms (GAs), a user enters the service data element (e.g. ga) in the search field and all GAs registered will appear on the browser. This feature was used to implement how to query and view registered genetic algorithms. In the browser, there are also fields to search for fixed data using specific names or dynamic search that brings out many outputs. This form is customised to allow design experts to issue query and view search algorithms. The form also has query fields for enumerative and deterministic algorithms; however they were not implemented due to time constraints.

5.5.5 MODO functional requirements

This class describes the functionalities of the system. This includes seamless access to distributed resources, input/output, data management, model building, job submission, computation of fitness functions and running optimisation using algorithms. These functionalities are described in section 5.3. The other classes such as main domain, criteria definition, parameter definition and constraints definition described the model building process that has been treated in section 5.3.1. The main domain class enables users to select the domain that they intend to build a model. This research used the three case studies as its main domains.

5.5.6 MODO services

The four services provided namely mathematical model building, collaboration; optimisation and computation are also described as classes in Figure 5.25. Details of these services are discussed in sections 5.3 and 7.5.

5.6 Summary

This chapter describes the design of grid services for carrying out MODO activities and discussed some examples of existing service specification models. The design of the interfaces provides an easy to use GUI for MODO experts. A contribution to service specification documentation is identified at the end of the design process. The

next chapter will provide a framework and architecture for MODO applications using the designs and specifications in this chapter.

Chapter 6 - Service Framework and Architecture of DECGrid

The last chapter provided an object oriented design for MODO services. This chapter uses this design and limitations identified (see section 5.5) in existing service specifications of related problem solving environments to provide a novel approach in service specifications for MODO applications. The process of defining grid services and the variables that affect collaboration using MODO services are first identified. This process and variables form part of the Decision Engineering Centre Grid (DECGrid) framework. The rest of the framework is created based on the service designs in chapter 5.

6.1 Transition from design to actual framework

Now that the design is in place, an appropriate framework is necessary to implement the design. Most grid frameworks that are used for computationally intensive applications are geared towards an efficient allocation of jobs to resources to maximise throughput (Kumar *et al.*, 2009). This may be to add business value to computational resources or to get quicker results for research purposes. Either ways, scheduling systems, middleware and resource brokers play important roles. In spite of grid popularity among researchers and design engineers, managing and supporting the grid infrastructure is still difficult and so designing and coming up with framework that allows researchers and design engineers to work within problem solving environment that provides user friendly interfaces is a welcome idea (Wang and Jie, 2009). This research intends to propose a framework which addresses the process of defining grid services and the variables that affect collaboration. The design in chapter 5 discussed how Globus middleware, Condor scheduler and PostgreSQL database server will be used to provide computational and optimisation resources as services. In software systems, the normal procedure is to use the design to obtain the framework for implementation. The aim and objectives of this chapter will be stated which reflect the transition from design to the framework in this research.

The aim of this chapter is:

To propose a framework and architecture for implementing DECGrid

And the objectives of the chapter are:

- *To identify the process of grid service definition*
- *To discuss the critical variables that make grid services practically useful for MODO applications*
- *To use the design in chapter 5 to come up with a framework for the research*
- *To provide an architecture for implementing MODO services*
- *To state the contribution in the framework*

6.2 Process of grid service definition

Like any service, a grid service has its target users and grid service providers strive to meet the requirements of their users (requestors). The process of defining a grid service is guided by this principle. For example IBM (Hayes, 2008), Google (Dyer *et al.*, 2008) and Amazon (Weiss, 2007) started their cloud computing projects based on some initial market assessment of the demand in the grid community. The choice of target users determines whether or not the grid will serve research communities or commercial and business communities. This again will determine the features desirable by such communities. For example, research communities are often comfortable with the proxy security feature in Globus toolkit which delegates authentication and authorisation of resource access to users through third parties while the business community is not happy with this. They would rather use bipartite security feature which restricts the right of authentication and authorisation to resource owners without involving third parties. However, regardless of what target users a grid service is to serve, a generic procedure is adopted here and can be ‘customised’ to meet various types of grid services. The generic process is described in Figure 6.1.

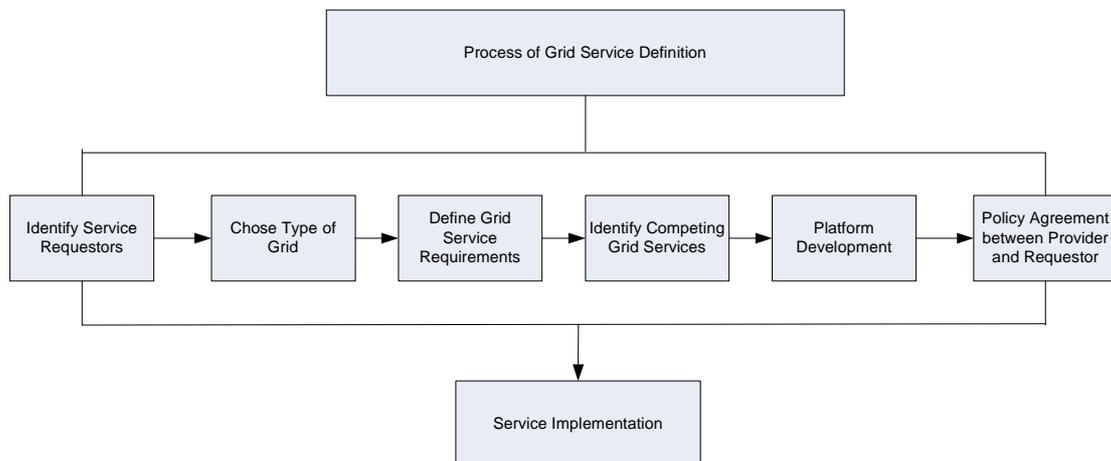


Figure 6.1: Process of grid service definition

Each stage described in Figure 6.1 is explained below.

6.2.1 Identification of service requestors

Like every utility, grid service existence and continuity is driven by users' demand. This forms the basis that grid service providers use to define and create services. The first process in grid service definition is to identify service requestors and decide on what type of services and requirements a service provider needs to satisfy. Service requestors range from the field of science, engineering and business and they may require services for data, computation, collaboration, optimisation and visualisation among others. Users may be from different companies and may be located in different geographical locations. These and many other considerations are captured during this stage to serve as decision guideline for the type of service provision (Kim *et al.*, 2008). This may help in delivering the required quality of service (QoS) on resources assembled dynamically from enterprise and service provider through abstractions and concepts acquired (Foster *et al.*, 2002a). In this research, the service users are design experts carrying out MODO tasks to obtain optimum solutions for designs. This category of users needs computational power, data repository, collaboration and visualisation capabilities within the services provided.

6.2.2 Choice of grid type to publish services

When service requestors have been identified, the provider decides what type of grid can accommodate the services the users need. For example, if users are mainly in the research and scientific domain, then a combination of computational and data grids which allows collaboration is a likely candidate. This is because researchers usually need computational power to perform some simulations and data could be generated which requires inputs from collaborating researchers to analyse. A single grid type may not satisfy users, so the normal convention as proposed by this research is to identify combination of grids that best suits users' needs. Grid types include computational grid, data grid, enterprise grid, extraprise grid and global grid (Abbas, 2001).

6.2.3 Definition of grid service requirements

Now that the requestors have been identified and the type of grid that suits them is also identified, the next stage is to define service requirements that will meet requestors' needs. Service requirements may include collaboration, dynamic pricing

and accounting system (Buyya, 2002), special security features, deadline and priority job completion, service discovery mechanisms, and communication protocols (Lin and Lin, 2006). Both functional and non-functional requirements are defined. In chapter 5, the design was based on MODO service requirements. Such requirements include the need to have a mathematical model building interface and to have a collaboration for design experts to collaborate. Other obvious requirements are the computation and data repository.

6.2.4 Identification of existing competing grid services

It is a good practice that the services provided conform to existing similar services so as to maintain a standard. This is done by studying the performance and problems faced by existing similar grid services with the aim of improving upon them and avoiding problems. During the literature review and industry survey in chapters 2 and 4 respectively, MODO PSEs were discussed. This gave the researcher an idea of existing PSEs in MODO domain and their capabilities (Kim *et al.*, 2008). The emergence of cloud computing is believed to have come out of the grid community to launch production grids for companies to see the business needs to invest in it even though the utility computing concept has been part of the grid concept (Buyya, 2002). The Geodise, FIPER, SORCER and DAME projects are the main competing PSEs in MODO applications.

6.2.5 Grid convention platforms development

This is one of the most important stages in grid service definition. Services do not operate in isolation; they interact with other services through grid ‘plumbing’ interface implementations that follow recommended standard interface protocols as provided by the Open Grid Forum (OGF). These protocols are part of the Open Grid Service Architecture (OGSA) and Web Service Resource Framework (WSRF) platform implementation that allow services to be dynamically created, named, discovered, used and destroyed without much configuration difficulties (Berman *et al.*, 2003). Every service provider that intends to join the grid follows the OGF recommended convention for developing the platforms upon which the services are published. Service requestors also need to do the same so that both interfaces (provider and requestor) conform to the same standard. This is what allows the service provider and service requestor to communicate effectively. This feature is partly a

technology issue and partly a business issue. Section 2.7.1.1 of chapter 2 describes the OGSA platform together with portTypes, GridService and service data elements (SDEs). Conventional grid services use these standards for services to communicate and be useful.

6.2.6 Policy and agreement between grid service providers and requestors

Service level agreement in grid services is increasingly becoming an important aspect to have multiple providers and requestors sharing and allocating resources and services for multiple applications (Wu *et al.*, 2009). Service providers and requestors usually have written agreement for the services to be provided along with functional and non-functional requirements. The service provider agrees to provide services that will deliver functions that satisfy some parameters such as QoS, computational power, data storage and management, security and optimisation to service requestors. Service requestors are charged for using the services on-demand basis as long as the services satisfy their requirements (Buyya, 2002). This agreement is necessary to make grid service stakeholders (providers and requestors) know their obligations. This is one of the features this research intends to incorporate and emphasise in the service specification document.

It is worth mentioning here that the legal framework for service level agreement has not been addressed in the service specification document in this research. This is considered as a future work.

6.2.7 Grid Service definition implementation

When service providers and requestors finally agree on terms of service provision and service requisition, the entire service definition is set for implementation. Every single grid resource (service) is uniquely registered and is uniquely traced to a service provider by a Grid Service Reference (GSR). This makes implementation of similar services by different service providers possible within a global grid. Also, every single instance of a service is uniquely identified globally by a Grid Service Handle (GSH). This property makes it possible to create multiple copies of the same service by different service requestors globally through stateful representation (Chao *et al.*, 2006). This research uses the factory pattern model for developing services which

uses the OGSA. Globus middleware is built on OGSA and has capabilities for GSR and GSH.

It is important to note some differences in what ‘grid service providers and requestors’ stand for from two different points of views. The first version is looking at grid service providers and requestors as ‘organisations, companies, institutions or individuals’. For example a university that builds grid services for its researchers (requestors) to use. This is the context in which this research uses the phrase ‘grid service providers and requestors’ in the ‘process of grid service definition’. This is a practical view of grid service definition which looks at grid services in terms of economic models of supply and demand or supplier (service provider) and customer (service requestor) (Buyya, 2002; Papazoglou, 2003). However, the same phrase (grid service providers and requestors) can be used to mean software or hardware systems that implement grid service provision and acquisition. These systems are the autonomous agents for service discovery, service orchestration, service brokering, service scheduling, information service, resource management and service authentication and authorisation (Czajkowski *et al.*, 2001). This is a systemic view point of grid service definition and is used in grid service representation.

6.3 Service framework

A framework for MODO service specification is developed for this research based on of the process of defining grid services and the variables that make grid services practically useful. This is the framework upon which the services will be built upon. The framework consists of two parts. The first part describes the high level service specifications that MODO application interfaces provide for users. This consists of general process of creating grid services, optimisation resource/service publication and registry, functionality of services and component service interfaces. The second part consists of detailed steps of service specification for programmers to use. UML class diagram is used to describe the components of part two which has already been described in Figure 5.25 of chapter 5. Figure 6.2 is the diagram that describes the framework.

6.3.1 MODO grid service framework

Equipped with the knowledge of different models in grid service architecture, this research has proposed a framework to be used for MODO applications. The framework is developed by capturing MODO requirements in literature and industry. This process was discussed in section 3.4 of chapter 3 (also see Figure 3.2). The challenges of related problem solving environments for MODO and their service specifications were identified and this also helped the researcher to come up with a framework that suits MODO. The framework begins with the process of providing grid services as outlined in section 6.2. This structured process of defining a grid service by a provider highlights the novelty of the framework which takes into consideration the requirements of users, other competing services and service level agreement. Including this process in the framework ensures reliability for services provided as users' requirements are captured and service providers' responsibilities are clearly stated in the service level agreement. The framework also provides a graphical user interface that takes the optimisation engineer step by step through the process of obtaining a model. The graphical interface ensures usability of the system for designers. The intuitive process of building the model is derived from information obtained from academic and industry experts. Access to resources is controlled within the PostgreSQL database server administrative file as well as the Globus grid resource allocation management (GRAM) gatekeeper and job manager. Figure 5.25 in chapter 5 described the service specification document that is used to come up with the framework. Figure 6.2 is the framework which gives the high-level generalisation of the functionalities that support MODO. Optimisation resources are published in the Globus WebMDS (Web Monitoring and Discovery Service) by the service provider so that optimisation engineers can search and view for resources they need. They can then decide to subscribe for the services that meet their optimisation requirements. Services are registered in the MODO Registry and are found through the `findService()` method (numbered as 1) and each service has a unique identifier called the Grid Service Handle (GSH) (numbered as 2). GSH enables many design experts to invoke multiple instances of the same service without any conflict. The services use the open grid standard service interfaces described in chapter 2. To make the services provided stateful, open grid services architecture defines a mechanism to expose a service instance's state data called `serviceData`. The concept is the same as declaring attributes in object-oriented (OO) programming languages. The data can be used for

reading, writing or subscription. Web services description language (WSDL) which is used for providing the service schema is used so that operations and arbitrary state information can also be made publicly accessible to both computational and optimisation services. The base GridService portType (from which all Grid Services inherit) defines operations for accessing the serviceData by name. One important attribute of serviceData element (SDE) is mutability. This means that the SDE value can change over time and is important for clients to understand the state changes of Grid service being accessed.

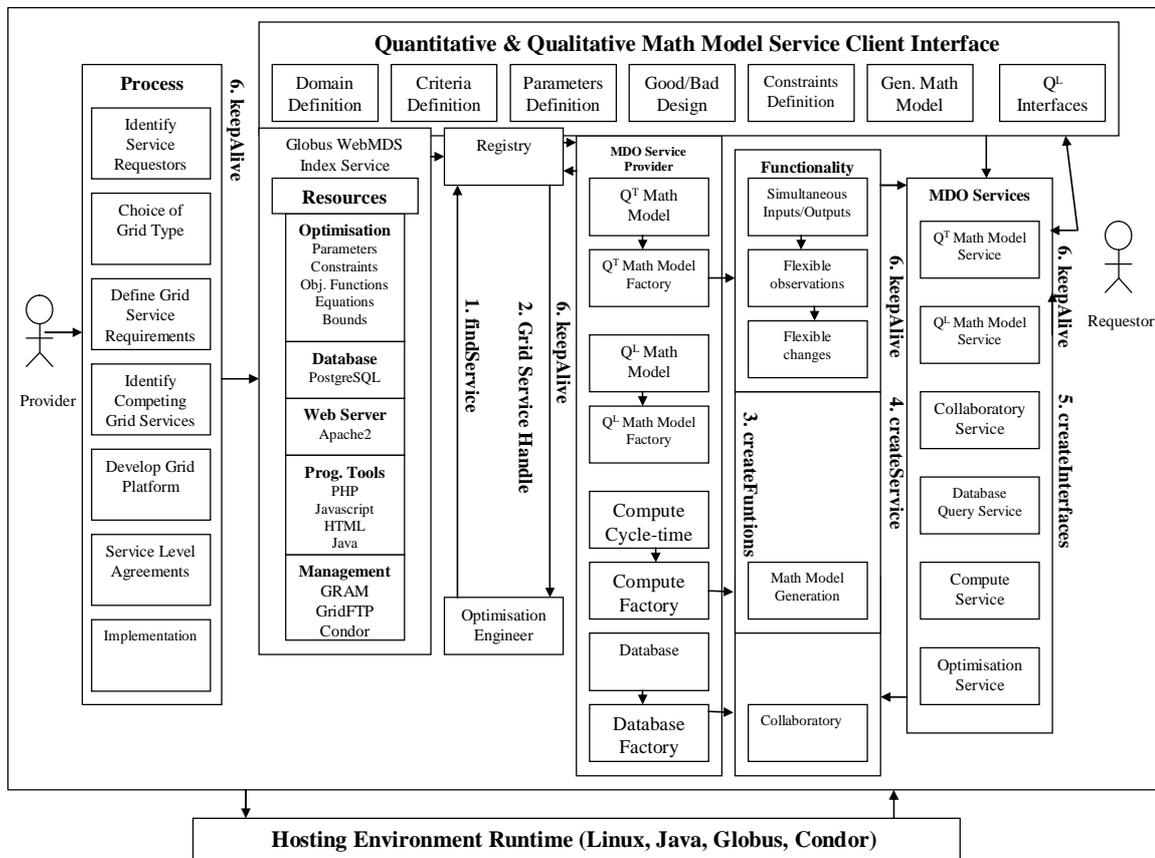


Figure 6.2: DECGrid MODO service framework

In Figure 6.2, the MODO service provided consists of quantitative (Q^T) model, qualitative (Q^L) model, compute cycle-time and database. Each of these has its factory where instances (copies) are generated. A functionality specification consists of simultaneous inputs/outputs, making flexible observations and changes in the parameters, collaboration and model building activity. Functionality are triggered by the createFunctionality() method (numbered as 4). An instance of a service or resource is kept alive by the Globus keepAlive() method (numbered as 6). MODO

Services component consists of Q^T Math Model Building Service, Q^L Model Building Service, Collaboration which consists of Database Query instance, Compute Service instance and Optimisation Service. The createInterface() method (numbered as 5) creates the interfaces that the optimisation engineer (user) interacts with to provide inputs as well as receive outputs. The main interfaces are the Domain Definition, Criteria Definition, Design Parameters Definition, Good/Bad Designs Definition, Constraints Definition and Model Generation Interface.

The functionality of the specification runs on Linux, Globus, Condor and Java runtime environment. Models are run on the basis of explicit or implicit characteristics as described during the design stage in chapter 5. The operations findService, Grid Service Handle, createFunctions, createService, createInterface and keepAlive are numbered 1 to 6 because that is the sequence in which the actions occur.

Figure 5.25 in chapter 5 shows the interaction between service providers and service requestors. Service-level agreement is the first step in the interaction. This is important for both parties to understand their responsibilities. As said before, Figure 5.25 describes more specific details of the DECGrid functionality and implementation platform as a specification document which is usually lacking in most grid deployments. Distributed optimisation engineers collaborate to generate mathematical model, submit computational jobs and make queries and observations. The framework adopts the loosely coupled portal for entry into the grid service using Globus Toolkit (GT). This means that the model is used when computational processes can be subdivided into independent processes and their results later on used for final processing or analysis. This framework is similar to parallel distributed model except that in this case, each node has its resource manager (Globus Resource Allocation Management) that manages resources and not a centralised job manager. The job manager is required to track the states of each completed and failed process. Condor-G is a good job manager for the client side to facilitate submission of jobs. The Condor server is usually started by the user and then jobs are submitted to this user job manager. The manager coordinates the refreshing of the user proxy that a grid resource must have in order to run the user's jobs. The user keeps supplying new proxies to the Condor manager at an interval of time. This means that the manager must stay alive while

jobs are running on remote grid resources to keep track of running jobs up to completion. Condor-G has the capability to recover from both the client and server side crashes. Globus GASS (Grid Access Secondary Storage) is used on the client side to manage the default data movement for the jobs that are running. MODO applications take advantage of the model to enable efficient coordination of changes in design parameters and the increase in computational power derived from the distributed computational synergy obtained from idle systems using Condor.

6.4 DECGrid architecture

From the specifications in Figures 5.25 and 6.2, the DECGrid architecture is obtained. Figure 5.25 identifies the steps for service provider to register optimisation resources using the MODO Resource Registry. Each node has a resource registry from which distributed users contribute resources. The aggregated resources are held in the MODO Aggregator Source as explained in section 5.5. The steps to build mathematical model using mathematical model building service is provided in Figure 5.25. Figure 6.2 provides the steps that a service provider needs to follow to actually implement MODO services. The interfaces that designers need to create models and enter optimisation parameters are also described in Figure 6.2. The interaction between providers and users and their responsibilities are the first stage described in Figure 5.25. This description forms the foundation of DECGrid implementation. Figure 6.3 is the diagram of DECGrid architecture. DECGrid consists of the server side and client sides and is run in Linux (CentOS 4.0) environment. Both the server and clients have Globus Toolkit 4.0.4, Condor, Apache and PostgreSQL-7.4 installed on them. The server side data management is implemented using PHP-5.2.5 and HTML (hypertext mark-up language) and client side in JavaScript and HTML. The server provides MODO services through the MODO Service Factory using the factory pattern model that is built in OGSA (open grid services architecture). The services provided are mathematical building service which consists of Q^T mathematical model building service and Q^L mathematical model building service. There is also collaboration service which consists of storage and compute services. These services are exposed (published) on the Globus WebMDS. WebMDS is the service in Globus which enables distributed users to view available grid services and their properties. Globus uses OGSA (Open Grid Services Architecture) standards for services and OGSA needs to have stateful services. This stateful property is provided by WSRF

(Web Services Resource Framework). SOAP (Simple Object Access Protocol) is used as the standard protocol for exchanging and sharing information among the distributed design experts across the grid network.

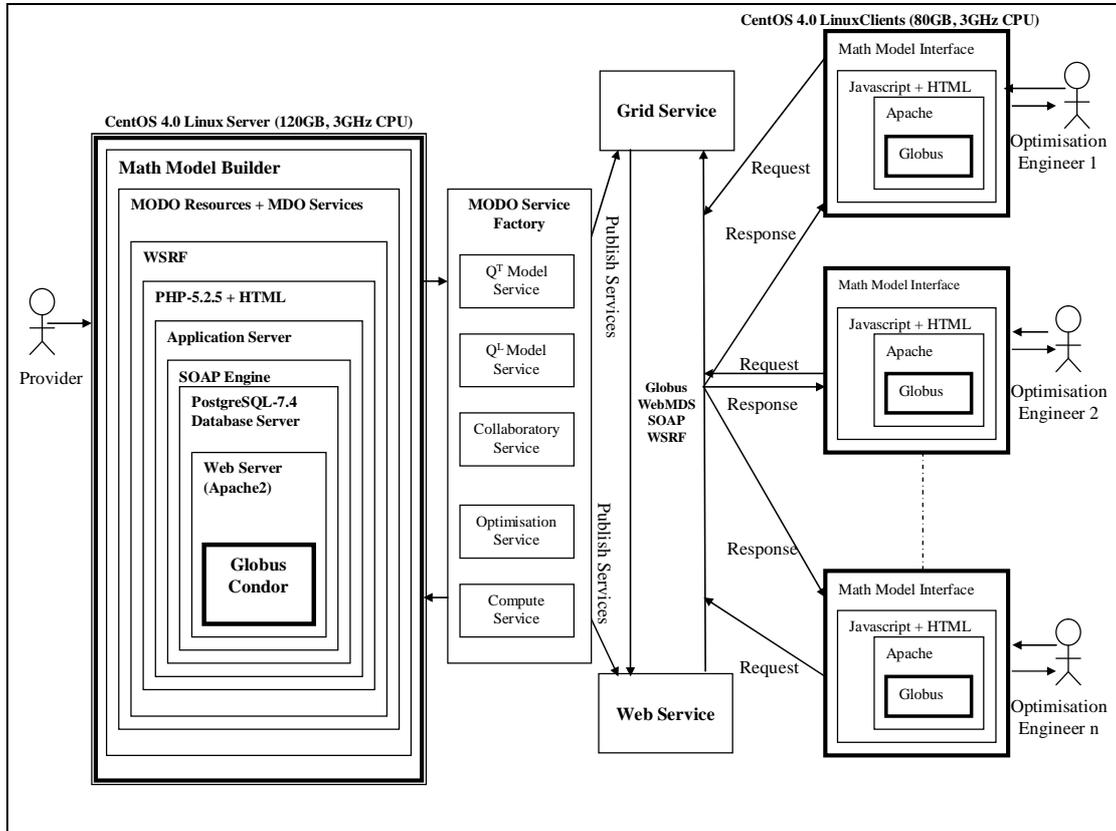


Figure 6.3: DECGrid architecture

Requests and subscriptions for services are made by requestors (optimisation engineers) through a web-enabled grid service infrastructure hosted by the server. Model building interface is presented to engineers to make inputs, collaborate and get outputs. Optimisation engineers have the flexibility to make observations on inputs made by other engineers and changes and amendments can be effected through the collaboration service and storage service. Condor scheduling system enables designers to submit jobs to idle nodes of other users to speed up the optimisation process. This capability is used to ensure computational throughput among collaborators. The result of this computational throughput is demonstrated by using one node to run optimisation process for the three case studies. Figure 8.19 in chapter 8 is the result obtained from this demonstration. Distributed designers make request for optimisation resources as they collaborate securely.

6.5 Research contributions of the framework and architecture

The service specification document (Figure 5.25) and proposed framework (Figure 6.2) and architecture (Figure 6.3) are to be used to implement grid services for MODO applications. Apart from the application aspect, the research contribution is very important. One of the research issues is the inclusion of service specification document for end users of MODO services which provides a structure for SLA between providers and requestors apart from the specifications for technical grid users such as grid programmers. This documentation is the information on the grid as a utility infrastructure. Another research contribution is the process of service definition for MODO interfaces. The way optimisation is carried out by scientists is now translated into interfaces that allow MODO experts to interact and visualise results. This intuitive research platform for optimisation engineers has the potential to trigger more innovative and creative thinking during math model building process (Gero, 1994). This is because the design engineers do not need to worry about how optimisation is performed as the system guides them through the optimisation process. Engineers now have more time to think about improving mathematical models rather than how to do the optimisation. The architecture that supports the service is also part of the research. The architecture makes provision for nodes to specialise in certain services. For example some nodes provide qualitative optimisation services while others provide quantitative optimisation and computational services. This is another way of using the grid to optimise the sharing of limited resources among researchers and companies as these services are contributed by different users of the grid.

6.6 Summary

This chapter presents the framework and architecture that will be used to implement grid services for MODO applications. Service specification document is proposed to identify grid components that make the grid a utility infrastructure. The research contributions surrounding the framework and architecture were identified. These framework and architecture will be used to implement DECGrid services in the next chapter.

Chapter 7 - Implementation of DECGrid and MODO Services

Chapter 6 provided the framework and architecture that are used to implement the interfaces and MODO services of DECGrid. This chapter describes the practical activities involved in implementing these interfaces, services and the interacting scenarios of MODO users. These activities range from grid infrastructural deployment, programming of case study interfaces and functionalities and MODO services implementation.

7.1 Preparatory statement

The building blocks to implement the services mentioned have been identified in chapters 5 and 6. Depending on the motivation for building and implementing grid systems, the level of preparation for implementation may vary from simple desktop grid or campus grid to complex global grid (Wells, 2008). This is identified in the design, framework and architecture development stages. However, whether simple or complex, the vision of all grid infrastructures is to scale towards accommodating decentralised administration for large scale computational resource sharing tasks (Silva, 2006). The service specification document in Figure 5.25 and framework described in Figure 6.2 are required for the actual implementation of the MODO services in this research. This is because both figures have the details of programming specifications and service functionalities for optimisation tasks. The architecture in Figure 6.3 is the physical arrangement of the computers that make up the DECGrid. Grid architectures determine how the system will deliver dependable services to distributed users across decentralised administrative domains (Wang and Jie, 2009). The DECGrid architecture ensures decentralised administration by deploying Globus GRAM on each node as resource manager for all resources on every node. This means that if one node fails, other nodes can still work and can discover the resources of the failed node when it is reactivated. This is an important feature for MODO experts who share resources such as computation, data and mathematical models. The WebMDS shows all available resources to all participating MODO experts. This section will now state the aim and objectives of the chapter and proceed to discuss the implementation activities.

The aim of this chapter is:

To use the design, framework and architecture of DECGrid to implement MODO services for this research.

The objectives of the chapter are:

- *To describe the tools used and reasons for using them in the implementation of DECGrid services*
- *To describe the process and stages of the implementation*
- *To implement MODO services and interfaces that are used for validation*
- *To discuss generality of application of DECGrid and implementation issues*

7.2 Facility and tools

Information obtained during literature review and industry survey helped in identifying the tools that make up the DECGrid facility. Grid projects such as Geodise, DAME, Gridbus and FIPER use similar tools. Another source of information is the Cranfield HPC (high performance computing) Grid which manages the Linux based computational facility that most science and engineering researchers use to submit FEA (finite element analysis), CFD (computational fluid dynamics) and optimisation computations for processing. The researcher is also on the mailing list of the Cranfield HPC facility. DECGrid is run in CentOS 4.0 Linux operating system (OS). The choice of Linux OS is necessitated by the fact that majority of the grid community uses Linux for grid implementation. The researcher initially tried using Windows OS to deploy Globus but on encountering problems tried to get solutions from the users' forum but many users were not familiar with problems under Windows environment. This made the researcher to take a decision on which OS to use for deployment. This means that in grid research deployment, collaboration with other researchers may help to avoid problems by identifying popularly used platforms by the community (Ong and Jiang, 2004). This further buttressed the findings during industry survey where 63% of grid users said they used Linux OS for grid deployment. Globus Toolkit 4.0.4 (GT4) is used as the grid middleware for deploying MODO services in the grid cluster. GT4 building blocks are built on the layered architecture which runs on OGSA standard and protocols. The transfer of huge MODO data is efficient using GridFTP which also has capability for stripped storage devices (Asiki *et al.*, 2009). Gobus is the most popular middleware among researchers

in universities as discussed in literature and industry survey. The scheduling system used to ensure computational throughput for optimisation processes is Condor 5.0 and PostgreSQL 7.0.0 is used as the database server for the management and reuse of MODO design parameters and variables. Condor is used with Globus known as Condor-G and has enhanced workflow management capability in the Directed Acyclic Graph Manager (DAGMan) feature which can intelligently differentiate between different computational jobs and data placement tasks in heterogeneous grid resource sharing systems (Kosar and Balman, 2009). This is in addition to the cycle stealing concept that ensures optimum utilisation of idle processors. PostgreSQL has capabilities for allowing efficient queries through secured protocols of Globus. The programming languages used for implementing different functionalities of the system are C, C++, Java, Javascript, HTML and PHP. C is used to implement the quantitative mathematical models for the 3 case studies; C++ is used to implement the qualitative model of the third case study (design of a manufacturing plant layout) in Qt Designer (Qt Designer is an open source object oriented graphical user interface design application software); Java, Javascript, HTML and PHP are used to provide web-enabled user interfaces for optimisation users.

7.3 Setting-up the grid infrastructure

A room was dedicated for setting-up the grid facility that served as DECGrid within the Decision Engineering Centre (DEC) at Cranfield University. Because it is the first of its kind within the centre, it was named after the centre as DECGrid. There are 8 computers each having 200GB hard disk, 3.00GB RAM and 3.20 GHz processing speed. The systems are dual boot from Microsoft Windows XP Professional or CentOS 4.0 Linux OS. One of the systems is made the server and the remaining 7 served as the clients. The grid deployment is done in the Linux OS only. This is for the same reasons explained in section 7.2. The operating systems run on different disk partitions in the ratio of 120 GB to 80 GB for Windows and Linux respectively for the server and 140 GB to 60 GB for the clients. The bigger Linux partition for the server is to accommodate service orchestration to the client machines (Foster *et al.*, 2002b).

After the Linux deployment, Globus Toolkit is installed on the server first. The certificate authority (CA) used is the SimpleCA. The grid security configurations were made to allow my-proxy authentication for other clients with authorised CA key

and passphrase (the name for long password in grid systems). This has the advantage of allowing remote grid resource management (Sacerdoti *et al.*, 2004). Globus has capability for cross-functional CA; that is multiple CAs can be used by different MODO users and can still communicate and share resources. File transfer protocol (FTP) in GT4 known as the GridFTP is configured and used to remotely transfer configuration files to the clients. The same GridFTP plays a key role in ensuring that optimisation data is securely transferred and shared among distributed experts using DECGrid. Metadata generated from optimisation processes are more efficiently and securely transferred to distributed grid nodes by GridFTP for reuse and share among optimisation engineers working from remote locations (Eres *et al.*, 2005). A component called RFT (reliable file transfer) in GT complements the capability of GridFTP by managing the transfer of data and files to different nodes. With GridFTP and RFT in place, the GRAM (grid resource allocation manager) is configured to manage the computational grid resources. The GRAM uses the JobManager to submit and manage jobs and GateKeeper to ensure that only authorised users' jobs are submitted and allowed to run on nodes. This capability ensures more reliability, consistency and integrity of optimisation data used by experts who are sharing design model, parameters and variables. The rest of the client nodes were also configured as the server. The certificates created by the SimpleCA in the server are copied to other clients and GridFTP, RFT and GRAM were configured as discussed above. The last activity on GT deployment is the configuration of web services which enable users to start the container that refreshes and updates optimisation resources that are published as services.

Condor scheduler is deployed to make computational resources available to remote users when owners of the resources are not using them. The resources used in DECGrid are the computational processing powers, disks storage of the 8 systems. This allows optimisation jobs to be received by one condor pool and executed by another idle pool. Globus GRAM is used in this research as an interface with Condor to run optimisation services which use Condor ClassAd to match requests for resources as they are made available. This process ensures computational throughput of grid resources and avoids wastage of computational cycle time. The server was made the condor master and can monitor the utilisation percentage of the pool and 7 other systems were made the client so that jobs can be flocked to them for processing.

There are 2 pools each with 3 computers. PostgreSQL 7.0.0 database server is also deployed on the 8 systems. To access data from any of the systems, an authentication process is required on each of the machines.

MODO on grid will also enable the exploitation of individual MODO services on per-use basis. The optimisation techniques discussed in chapter 2 are all geared towards obtaining optimum designs that can increase the quality products manufactured at lower cost and ultimately increase the profit of the company. Though these techniques are there, what is lacking is the problem solving environments that allow efficient collaboration for MODO experts to work together (Lee *et al.*, 2009). This is why a framework must align with the needs of a company. The needs of MODO applications consist of collaboration, computational power and data storage. The distributed resources are managed uniquely and globally through the GIS (Grid Information Service), GRIS (Grid Resource Information Service) and GRIIS (Grid Resource Information Index Service).

7.4 Case study considerations

The implementation process takes into consideration the case studies that are used to validate the DECGrid. The case studies are optimisation of gas turbine blade cooling system, the welded beam problem and the design of a manufacturing plant layout. The first consideration is the service interfaces that allow generic MODO tasks to be performed by distributed experts. Although the interface in this case allows the selection of a problem domain for optimisation, this is just to limit the search to the optimisation resources needed in each domain. However, the system has capability to add more application domains to cater for generality. The grid is an infrastructure that allows seamless access through the grid portal to virtualised resources (Kacsuk *et al.*, 2004). DECGrid uses this quality of grid using Globus services that allow different domain users to access the same resources from distributed locations. The case studies involve multi-disciplinary collaboration in which parameters and variables are obtained as workflows from different collaborating experts and this makes the feature of grid portal very important to carry out multi-objective optimisation for different domains using shared resources. This provides a generalised platform upon which MODO applications can be run to solve diverse problems. Figure 7.1 is the problem domain definition interface where the 3 case studies can be selected on any of the

nodes to run optimisation on the entire grid. Different domains can be selected at the same time on different machines and still be processed concurrently using shared computational resources which are scheduled by Condor.

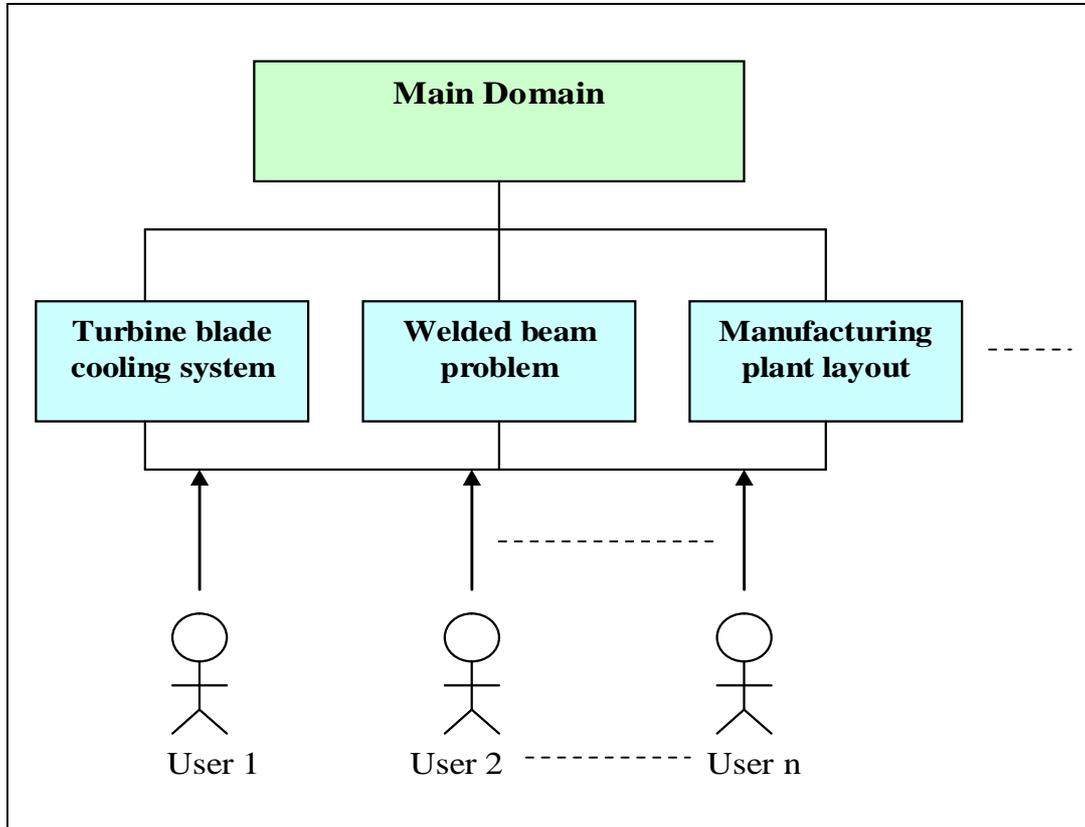


Figure 7.1: Main domain selection interface for distributed users

7.5 Services implementation

Now that the tools to be used are identified and the system is set up, the different services that will deliver multi-objective optimisation processes are implemented. The implementation is based on the OGSA (Open Grid Services Architecture) and WSRF (Web Services Resource Framework) standards which are part of Globus middleware. The services expose high-level interfaces to optimisation experts as easy to use graphical user interface (GUI). This feature is termed as component optimisation service implementation for MODO applications.

The DECGrid architecture in chapter 6 makes use of the grid service data elements (SDE) to present the properties of optimisation resources as MODO services. SDE is the extensible mark-up language (XML) feature that defines the properties of a

resource. SDE can be static or dynamic (Atkinson, 2003). The SDE are described in the XML schema that implements the different MODO services whose life-time management is done by uniquely identifying each service instance by an element called grid service handle (GSH) and are supported and held by the grid service reference (GSR). This enables the MODO services to be invoked at the same time by different collaborating users without much interference as each service instance has its state and life time managed differently through the GSH and GSR. As has been said before, the important grid service interfaces (portTypes) that implements the interfaces are the GridService, NotificationSource, NotificationSink, Registry and Factory. The GridService portType uses the FindServiceData to query optimisation resources and is responsible for service discovery, NotificationSource is notification of the instances of services to service requestors, Registry registers or deregisters services and Factory creates service instances using CreateService operation. Before a service can run, the Globus container must be started. This shows all the available services running.

Component optimisation service (COS) is a service that presents high-level information on how to build multi-objective optimisation models and provide visualisation, quantitative, and qualitative model building, optimisation and parameter inputs as services. The Globus Index Service presents optimisation resource property for request and subsequent subscription by requestors. It is an interactive and intuitive service that is aimed at helping novice design engineers who do not understand the working principles of optimisation to perform optimisation tasks. The design engineer however knows the domain in which to perform optimisation. The layout architecture of Globus enables different computational resources to be provided at different layers. The component services run on the application layer which is the layer that the end user interacts with. The network and connectivity layers allow the distributed service node and worker nodes to communicate through authentication mechanism of Grid Security Infrastructure (GSI). For example, one of the component service is the mathematical model building service (discussed in more details later) that allows the design engineer to enter properties required in the domain, and the Globus Trigger Service presents the designer with a component representation of the problem domain as described in Figure 7.1. The designer is prompted to define criteria, design parameters and constraints using graphical user interfaces. These component interface

representations are linked to different optimisation models in the Globus Archive Service. In NSGA-II, the `problemdef.c` file is where mathematical models are used for computation. In this research, this file is made available to the services through a PHP script and Apache.

All services in this research are implemented using the popular factory and instance patterns where resources are created by the factory service using the `CreateService` method and resources created are accessed using the instance service. This enables the grid system to run multiple MODO service instances for many experts at different nodes. Each instance created returns an endpoint reference (EPR) to the web services (WS) resource. EPR is a unique address for each WS-resource. Figure 7.2 shows the factory/instance patterns for MODO services in DECGrid.

The distributed clients in DECGrid have different MODO resources. Each node can invoke the `CreateService` method on another node that has the resources it needs and an instance of that resource can be created. These resource instances get published as resources or services in the Globus WebMDS (Web Monitoring and Discovery Service) which can be viewed from any node within DECGrid, signifying that GateKeeper has authorised the creation of an instance. With the factory/instance patterns, resources can be shared among MODO users from distributed locations and the workflow process of optimisation can be organised for reuse within the virtual organisation (VO).

This is possible because each node requests for resource instance from the factory service and the factory service in turn creates resources from the resource home. The instance service finds resource instances from the resource home and performs needed operations on the instances and the resource itself is also managed by the resource home.

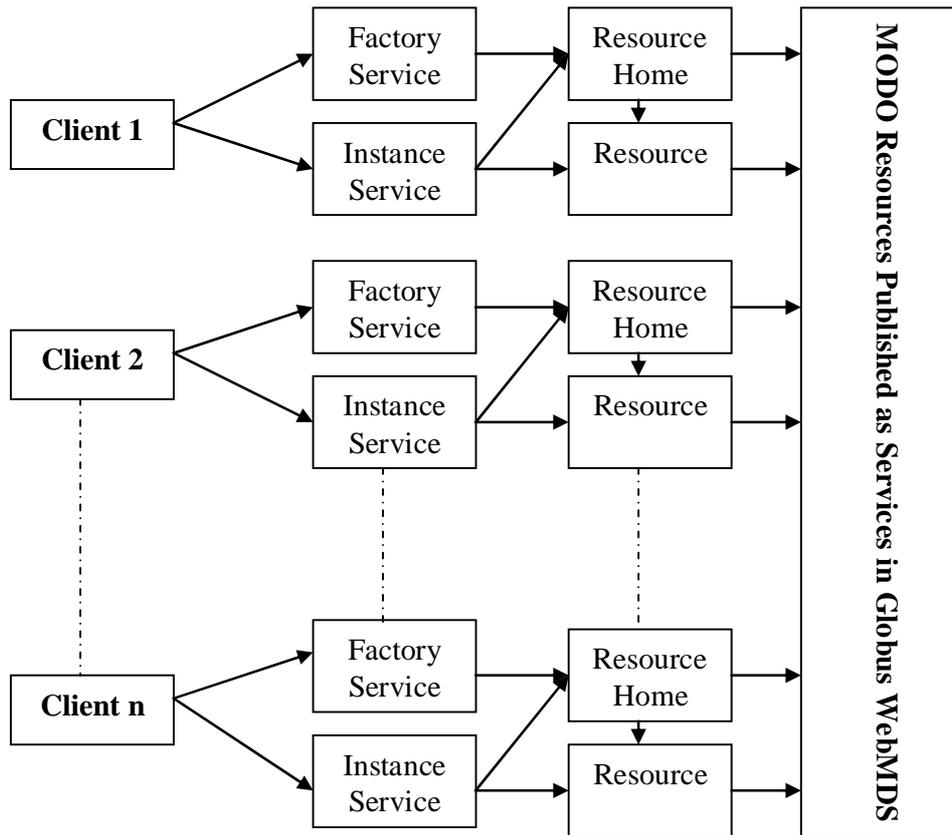


Figure 7.2: Factory and instance patterns of MODO resource creation

7.5.1 How to write a service

There are five major stages in writing the codes and executing a service. These steps are discussed for mathematical building service as an example. The first step in writing grid service is to define the interface. This can be done using java language or WSDL (Web Services Description Language). The interface definition is the information that the service displays to the user. The mathematical model building service has four variables that a user needs to build a model as explained in the design (chapter 5). Figure 7.3 shows the mathematical model interface definition. The portType (interface) is given the name MODOPortType and the properties are described in MODOResourceProperties. The properties consist of data type (integer, character, etc). The interface definition defines the variables that mathematical operations will be carried on them. Variables for input and output messages are defined so that messages can be sent to the user.

```

<?xml version="1.0" encoding="UTF-8"?>
<definitions name="MODOService"

targetNamespace="http://www.globus.org/namespaces/examples/core/MODOService_instanc
e"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:tns="http://www.globus.org/namespaces/examples/core/MODOService_instance"/>
<!--=====
                                M O D O   P O R T   T Y P E
=====-->
<portType name="MODOPortType"
  wsdlpp:extends="wsrpw:GetResourceProperty"
  wsrpw:ResourceProperties="tns:MODOResourceProperties">

  <operation name="variableInput">
<input message="tns:VariableInputMessage" />

      <output message="tns:VariableOutputMessage"/>
</operation>

  <operation name="leftOperator">
      <input message="tns:LeftOperatorInputMessage"/>
      <output message="tns:LeftOperatorOutputMessage"/>
</operation>
  <operation name="rightOperator">
      <input message="tns:rightOperatorInputMessage"/>
      <output message="tns:RightOperatorOutputMessage"/>
</operation>

  <operation name="outputVariable">
      <input message="tns:outputVariableInputMessage"/>
      <output message="tns:OutputVariableOutputMessage"/>
</operation>
  <operation name="getRelationship">
      <output message="tns:GetMathModelOutputMessage"/>
</operation>

</portType>
</definitions>

```

Figure 7.3: Mathematical model interface definition codes

The second step is to implement the service using java programming language. The codes for this can be seen in Appendix-II. The java program accepts the inputs and forms the relations of equations, constraints and inequalities to come up with the model. The third stage is to configure the deployment of the service. This is done in a file called the deployment descriptor using the Web Services Deployment Descriptor (WSDD). WSDD is the functionality that instructs web services and grid services on how to publish a service. The file is then deployed in the web service container. When the Globus container is started, this service is seen running among the services (see Apendix-III). This service is available to all client nodes that are connected to the master node and have their containers (Globus services) running.

The fourth step is to create the grid archive (GAR) file. This file is used to put together the interface definition, the java implementation and the deployment descriptor so that these separate functionalities can be presented to the user as a single service. The GAR file is created using the `./globus-build-service.sh` command in Globus. The fifth stage is to deploy the GAR file created into a grid service. This is done by using Globus and Ant (a package that unpacks files) to unpack the GAR files and deploy the service to the location desired. To deploy the service, the following command is used:

```
globus-deploy-gar $MODO_SERVICE_DIR/decgrid_mathbuildingservice.gar
```

This service can be undeployed using the `globus-undeploy` command.

7.5.2 Mathematical model building service

The steps for writing the service have been described above. This section will describe how the end user used the services. A web-based interface is provided for the user to enter those parameters described above to be able to build a model. When a user selects the domain, the mathematical model building process is the next action that is expected. Alternatively, if the model is a complex one and is available, then it can be uploaded. This service consists of two subcomponents namely the quantitative (Q^T) and qualitative (Q^L) model services. The Q^T optimisation service provides the interface for the building of mathematical models and then linking the math model to the NSGA-II optimisation code. The concept assumes that the designer does not know how to perform optimisation or how to choose the best optimisation model that suits his or her problem. However, the designer knows what is expected of the design to produce an acceptable product. The Q^T service guides the designer through a series of prompts for the model to be built based on the designer's responses. This grid service mathematical model builder is one contribution to conventional computational grid services which concentrate on improving computational power to speed-up the optimisation process with the assumption that the model has been built somewhere as a resource, waiting to be invoked.

The first step towards building the model is the invocation of the Globus `DelegationService portType`. This is important because the problem domain that the

designer may choose may not reside on the local grid node in use at the design location. The resources and expertise of the domain may be geographically far apart and the constraints, data and model may also be at different grid nodes. The DelegationService (DS) can be used to fetch resources from any side through the Globus single sign-on proxy capability and reliable file transfer (RFT) protocol. The next step is the prompt for the designer to supply problem domain. The problem domains are the three case studies described earlier as shown in Figure 7.1. An optimisation schema based on the Web Services Description Language (WSDL) is used to describe the data needed for MODO service as shown in Figure 7.4. This is the schema that displays information about the optimisation data (resource) published by a provider. It shows the date and time the resource is published. It also tracks the date and time a user uses an instance of the resource and the network information of the node that the resource is being accessed.

```
<MODOServiceProvider>
  <optimisationInformation>
    <optimisationData>
      < put optimisation data here>
    </optimisationData>
    <optimisationDateAndTime>
      < date and time of retrieval>
    </optimisationDateAndTime>
    <optimisationSourceURL>
      <URL where optimisation data is retrieved>
    </optimisationSourceURL>
  </optimisationInformation>
</MODOServiceProvider>
```

Figure 7.4: Optimisation data schema

As the problem domain is specified by the designer, the Globus NotificationConsumerFactoryService (NCFS) is invoked. NCFS notifies the most likely expert grid nodes to get ready for request from a consumer (requestor). The criteria definition for what to optimise is supplied by the design engineer. The

mathematical model building process is divided into explicit and implicit categories as discussed below.

7.5.2.1 Explicit math model building

Explicit models are mathematical models that are not too complex to build as explained during the design. These are simple mathematical formulations and constraints. The second and third case studies are based on explicit models. For explicit model generation, there are 4 generic fields namely variables, left operators (this can be (, +, -, /, etc) right operators (this can be (, +, -, /etc) and output variable/assignment. Figure 7.5 shows the explicit interface implementation.

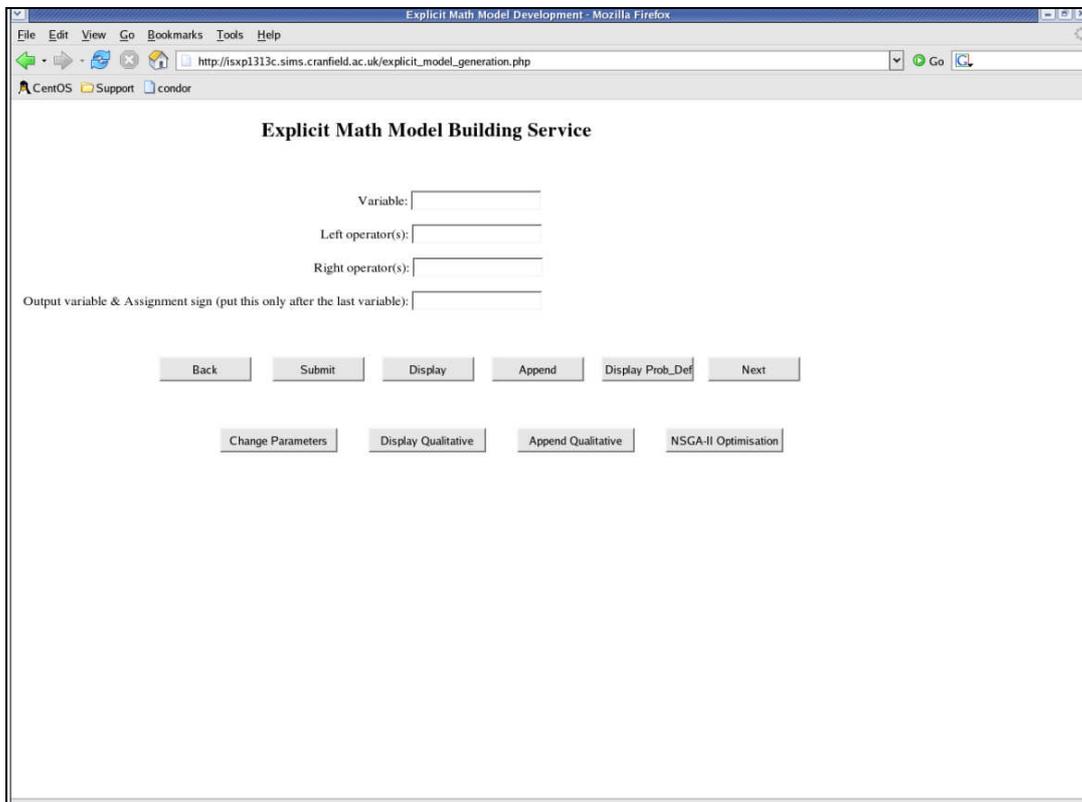


Figure 7.5: Explicit model building interface

The steps that were explained in the design will again be briefly described for clarity. A user will enter variable and/or left/right operators and submit it continuously until all input variables are complete for an equation/inequality. The last action to generate an equation in a model is to enter the output variable together with the equality sign (=) and submit. This will create an equation. This process is repeated again for the next equation until all equations/inequalities in the model are exhausted. When the model consisting of the objective functions and constraints are built, the display action displays the complete model for confirmation by the design expert. If the

expert is satisfied with the model, he or she can use the append functionality to link the model to the `problemdef.c` file of the NSGA-II. The NSGA-II optimisation functionality invokes the NSGA-II Input Parameter Optimisation Service so that the design parameters can be entered for the optimisation to be run. Various components of the model can be supplied by different experts from distributed geographical locations. For example the objective functions can be supplied by a different expert and the constraints by another expert. This is common with MODO applications in which multidisciplinary experts are involved in the model building process. This scenario will be demonstrated in the next chapter during validation of the prototype using the case studies mentioned.

In the design description, there are interfaces for entering the criteria (outputs), parameters (inputs), constraints and constants. These 4 fields described in Figure 7.4 are used to create instances that get the variables, criteria, constants and constraints. The input parameters are obtained using the NSGA-II input parameter optimisation service. The criteria are then linked to the input parameters and finally to the optimisation service.

7.5.2.2 Implicit math model

Complex and difficult mathematical models are tedious to build and are referred to as implicit models. These models usually require variables or equations that are dependent upon other variables and equations. Because of this, the mathematical model building process used to obtain explicit models above cannot be used. Instead the model is loaded from a file onto the NSGA-II model definition file. Figure 7.4 is the implicit mathematical model interface.

The first case study is an implicit model. The model is uploaded by using the upload model functionality and can be displayed on the browser using the display model functionality that is described in the case of explicit model. After this, the model is treated the same way as explicit model by linking it to the `problemdef.c` file of the NSGA-II. The next service allows parameter entry so that NSGA-II can use the parameters to run the optimisation. These interfaces are customised to suit MODO approach running NSGA-II on grid environment. The importance of building models is that multidisciplinary experts interact and the interactions result into decision

making on the components of the model which represent physical phenomenon of the problem involved (Sobieszcanski-Sobieski and Haftka, 1997).

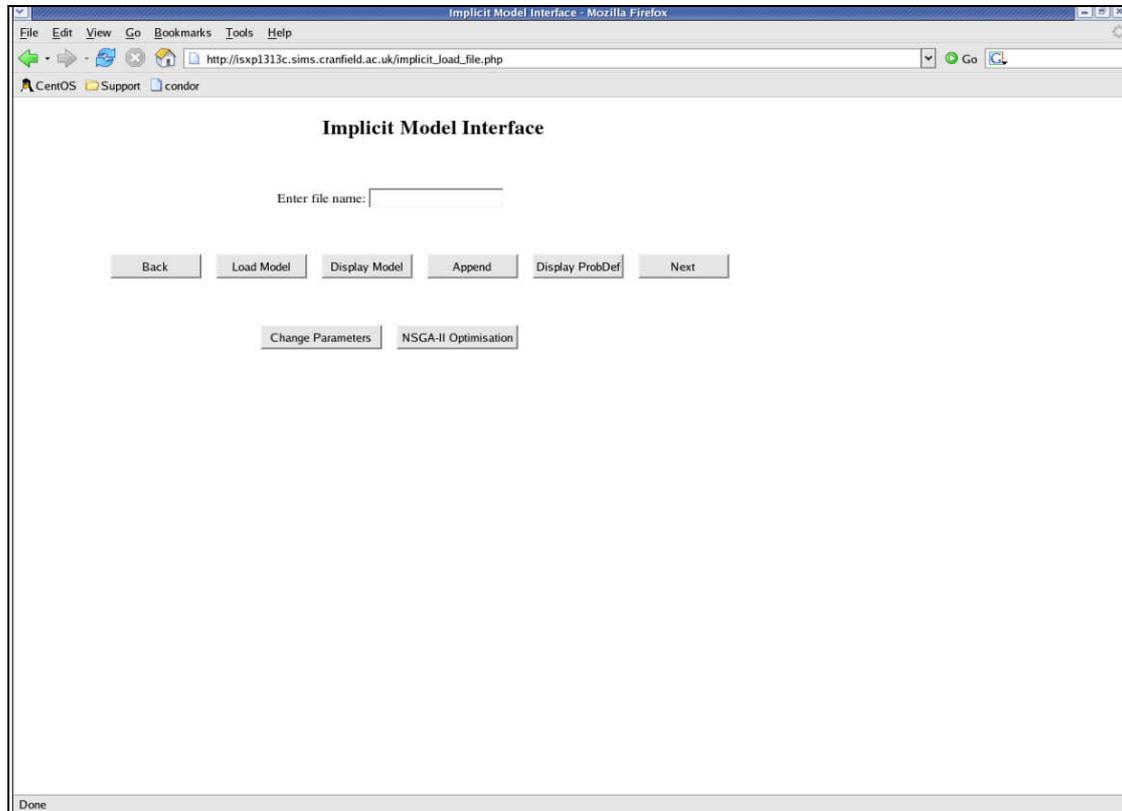


Figure 7.6: Implicit math model interface

7.5.3 Search strategy selection

It is important to use the appropriate search algorithm for every model. This can be done by providing the characteristics of the model to a query form. The properties could include number of objectives, types of objectives, number of constraints and number of variables and variable types (discrete or continuous). The query will match these properties to appropriate algorithms. Multi-objective design optimisation (MODO) is not aimed at finding a single optimum solution, but multiple optimum solutions known as Pareto solutions. Each of these solutions is known as Pareto optimum (Nebro *et al.*, 2007). The search strategy in this case is to find the Pareto front (graph of points joining each Pareto optimum solution). MODO uses three main search strategies namely enumerative, deterministic and stochastic strategies (Luna *et al.*, 2006). Enumerative and deterministic have finite population search space, though they can be computationally and data intensive after many iterations of computations.

Stochastic uses dynamic search population space. This research only provided NSGA-II algorithm which is a stochastic algorithm. However, the Globus search browser interface is configured to accommodate enumerative and deterministic algorithms in future research.

Using grid computing technology, a search strategy selection service based on Globus Web Services Monitoring and Discovery System (WS MDS) is developed to suit enumerative and deterministic strategies on one hand and stochastic strategy on the other hand. MDS is used to monitor and discover MODO resources such as optimisation algorithms that fall under the different search categories within a virtual organisation (VO). The Index Service collects monitoring and discovery information from optimisation resources and publishes it in one location for requestors to access. An intuitive and user-friendly web interface known as service group provides requestors, in this case optimisation engineers to view and query grid resources, in this case optimisation resources. This is done using the Open Grid Service Architecture (OGSA) browser graphical user interface. This browser allows users to query resources that are registered. The browser has pre-configured (fixed) and open-ended (dynamic) query capability. Pre-configured query searches for particular resources while dynamic query looks for resources that have dynamic or varying properties. The first stage of using the search strategy selection is to run the WebMDS and the browser appears with options to view all resources or particular resources using special query forms. There are basically four default query forms on the browser. Two are preconfigured queries and so are suitable for enumerative and deterministic search strategies while the other two are configured to be used for stochastic search strategy. Figure 7.7 shows the diagram of the search strategy selection query form. Although in this research, only NSGA-II is used for the 3 case studies, the search strategy selection (SSS) functionality is aimed at making the system robust enough to accommodate other algorithms.

Preconfigured queries are aimed at specific optimisation processes. The output is usually predicted based on experience. The search focuses on a determined direction. This strategy is suitable for enumerative and deterministic search strategies. Open Ended Optimisation Queries adopts strategy that searches optimisation resources in many sources at different grid nodes. These are suitable for stochastic search strategy.

The clients (optimisation engineers) submit queries to optimisation service registry (see Figures 5.16 and 5.25) to search for a search strategy. Clients intending to use stochastic strategy submit queries to view all the resources published through a single source or many sources. The service group also displays information of resources and users can submit queries through it.

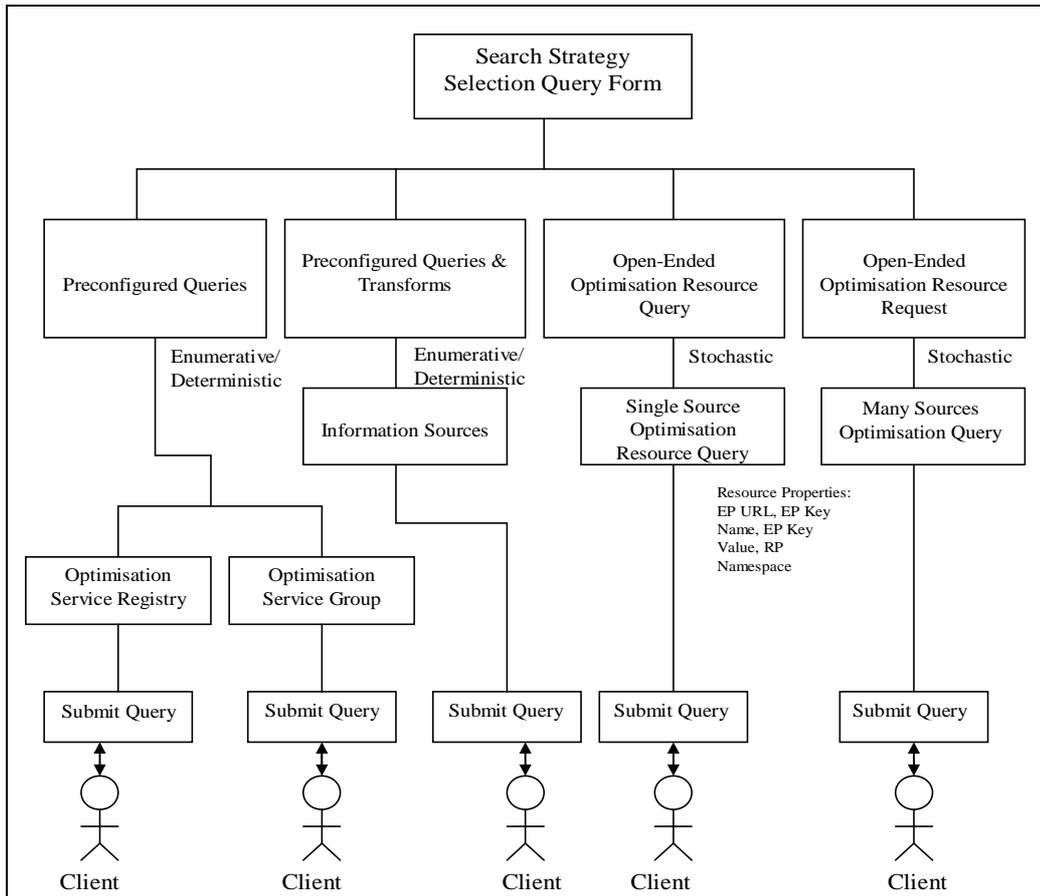


Figure 7.7: First stage of search strategy selection optimisation resource query process (end point =EP; resource property=RP; uniform resource locator=URL)

Figure 7.8 shows the search strategy selection procedure. Optimisation resources are first published by providers using the Globus RegistrationAggregatorSource component of the Aggregator Source service and then requestors (optimisation engineers) use the web service group to view available optimisation resources and determine which ones are suitable for their jobs. After determining the resources to be used, the Globus SubscriptionAggregatorSource is triggered for requestors to subscribe resources. The optimisation expert could invoke the ExecutionAggregatorSource to execute the search algorithm if the algorithm is linked

to other resources such as Globus Access and Secondary Storage (GASS) which may hold the design parameters and GridFTP (Grid File Transfer Protocol) which is used to transfer the input files. Alternatively, there are parameter and optimisation services that perform input parameter and optimisation tasks separately. The main task of the search strategy selection is to enable easy selection of a search strategy that best suits the optimisation problem in question. Each algorithm is identified by its properties (classical or stochastic), endpoint (EP), EP key, EP value or other resource properties (RP). End point (EP) is the identification tags for registered resources and queries are made by entering EPs in the browser to search for the resource. The optimisation can be split into quantitative (Q^T) and qualitative (Q^L) components which are handled by the Q^T and Q^L services respectively.

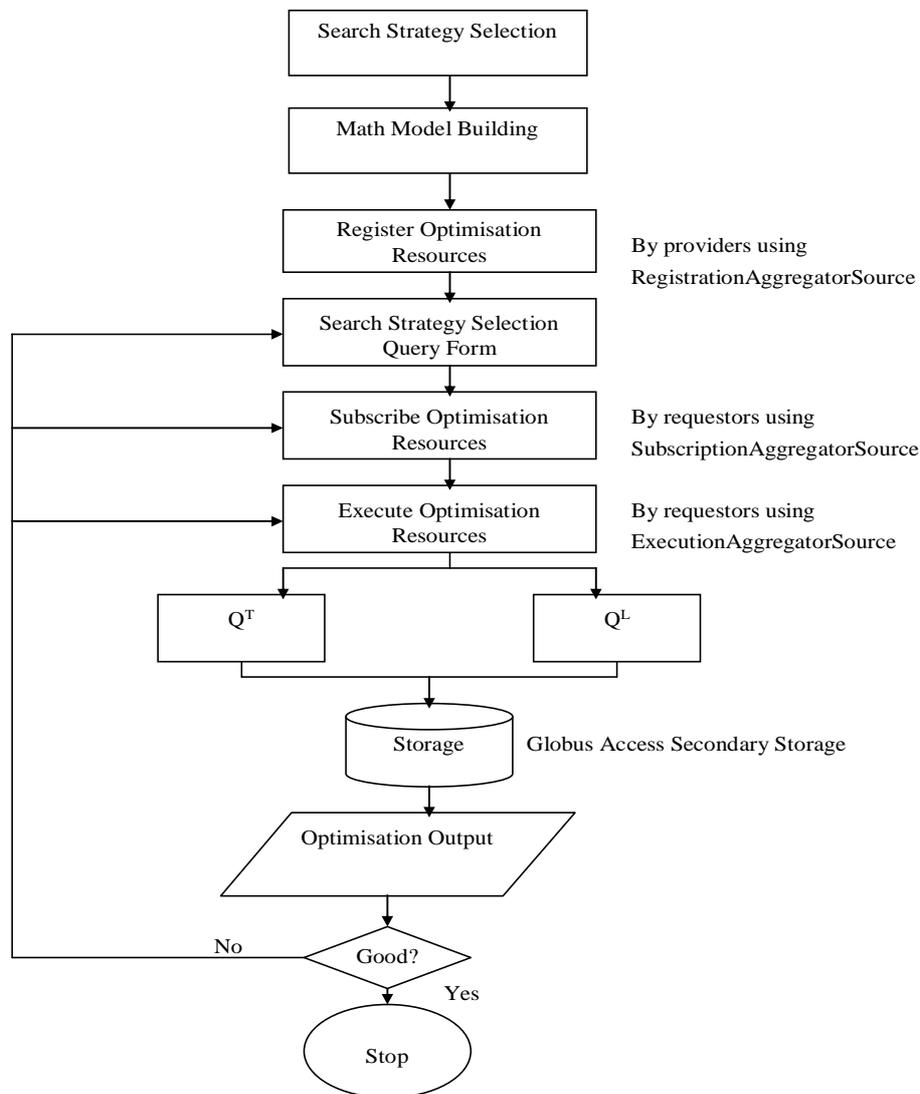


Figure 7.8: Search strategy selection process

The search strategies in the grid are accomplished by understanding the way services are provided using standard XML (Extensible Mark-up Language) schema. This is one of the objectives of the research (to understand grid environment requirements for MODO application services). This research used the Globus toolkit service schema (similar to Figure 7.4) to implement all the MODO services provided in DECGrid.

The advantage of this concept is that it allows optimisation engineers to search for the optimisation algorithms that best suit their requirements and the flexibility to try it over again with different algorithms at the same time. Search strategy selection interface allows viewing and querying of registered resources. The aggregator source (see section 5.5.3) accepts registered resources and allows subscription and execution of the services.

7.5.4 Collaboration service

Collaboration service is the term used for a grid system that implements collaborative sharing of grid resources and access to databases to accomplish a common goal within a virtual organisation. In this research the collaboration service implements information and computational capabilities through interfaces that allow optimisation engineers to share data and information for decision making. PostgreSQL database server is deployed on each node of the grid and different access rights are given to different users according to needs and trustworthiness. During RFT (reliable file transfer) configuration in Globus deployment, the nodes were configured to enable TCP/IP (transport control protocol and internet protocol) connections to the PostgreSQL databases. A database that allows trust entries for nodes was created on each node. This database was named modoDatabase (multi-objective design optimisation database) and trusted access was ensured by placing a line in the configuration file (pg_hba.conf) of PostgreSQL. This line consists of the host, database name, username and host IP address. This is the feature which makes the database server to allow optimisation engineers to collaborate by having access to the different nodes. Users that were created during Globus installations have their usernames and passwords embedded in the java naming and directory interface (JNDI) schema file (jndi-config.xml) of Globus to ensure that they can use the RFT and databases. By starting the Globus container, all new RFT configurations get loaded to refresh new users that are added and new database schema that are created.

These configurations ensure the integrity of optimisation data during collaboration as only authorised users have access to the database.

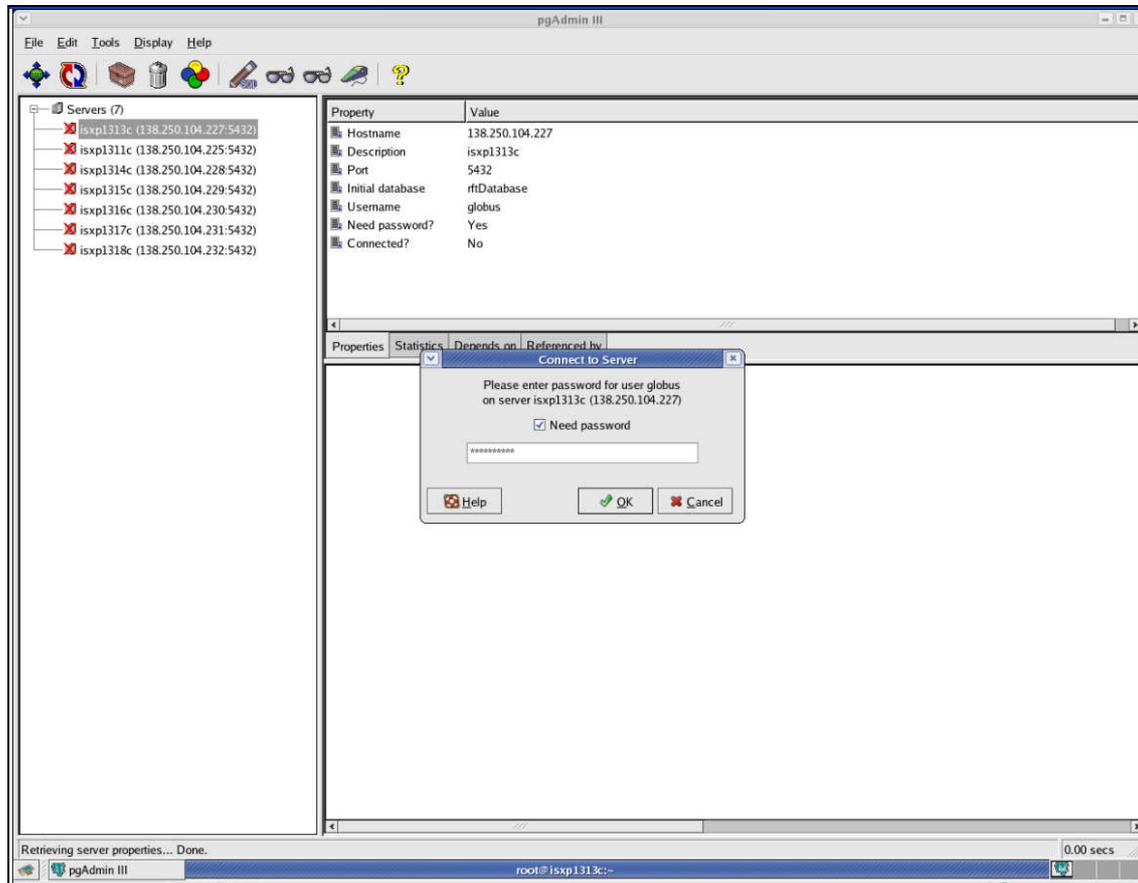


Figure 7.9: Seven servers display and prompt for password to have access to a host

The importance of collaboration in grid infrastructure is to provide a thin-client user access to computational resources that the user does not have through the grid portal (Russel *et al.*, 2002). This is common among computational and data driven applications in astrophysics, multi-objective optimisation and oil and gas seismic simulations. In this research, this concept is explored to use some of the clients as thin clients when the resources for optimisation are not on that node but the node needs them to carry some optimisation tasks. For example, in the case study for design of manufacturing plant layout, the capability for visualising the design by distributed experts which then allows the designers to rate the designs is not on all nodes. However, every node can have access to this capability as all resources are managed as virtual resources on all nodes.

The different data structures, data types and formats for the variables, constants and design parameters are also implemented in PostgreSQL. To get access to any of the servers (hosts), username and password are required for each host. This is in addition to the grid security feature. Figure 7.9 shows the servers.

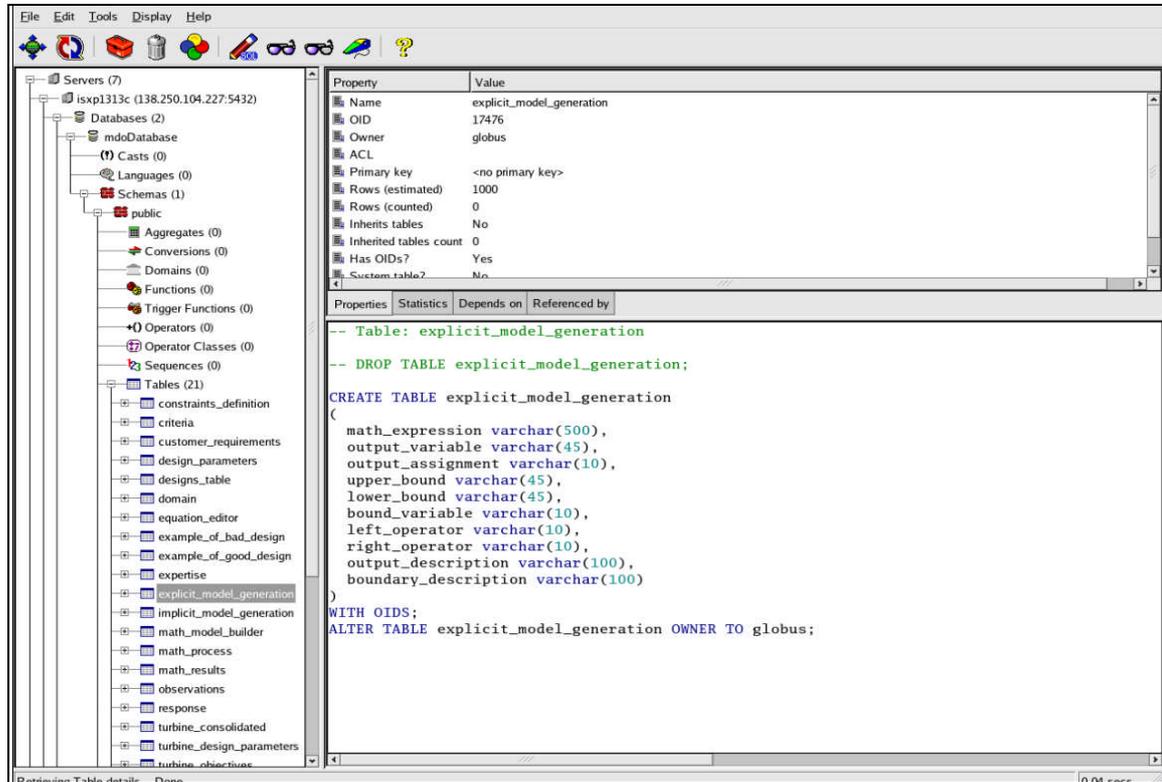


Figure 7.10: MODO tables in MODODatabase

If log-in is successful in the host, the database displays tables. Tables can also be created by users if they are given permission to do so otherwise they cannot create new database or table. Figure 7.10 shows the different tables that hold optimisation data. For example constraints are in constraints table and design parameters are held in design_parameters table. These tables are updated based on the restrictions placed on them. The idea of having the optimisation data stored in a database is to allow reuse of design parameters.

Apart from sharing data, the collaboration also enables the distributed users to share processing power, disk storage and memory for efficient computation of design optimisation tasks. Figure 7.11 shows how the virtual server with data server and compute server present their functionalities to other nodes to use instances of these capabilities as virtual resources. The clients A to D use a common grid portal either to

access data on PostgreSQL database server or to send some computations on the compute server. The concept of using Condor-G to send optimisation jobs for efficient throughput will be discussed later. Though virtual machines have been used before the concept of grid computing came, grid services provide large-scale abstraction layers for computational resources to be shared at lower overheads and functionalities that simplify problems that are inherent with centralised systems (Figueiredo *et al.*, 2003).

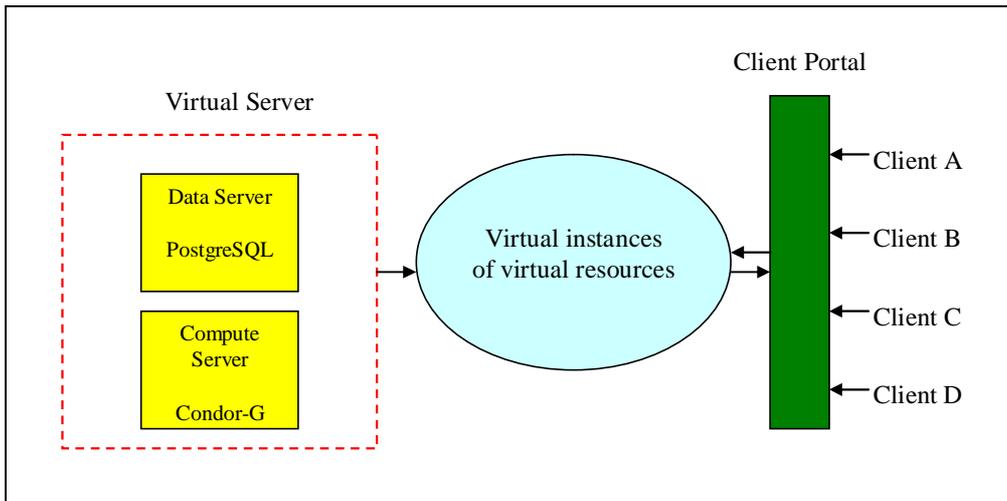


Figure 7.11: Virtual sharing of computational resources

7.5.5 NSGA-II input parameter and optimisation service

At this stage it is assumed that the mathematical model is ready and the search strategy that suits the model has been chosen from the WebMDS based on the model's characteristics. The MODO experts working on distributed nodes on the same problem can share resources through the collaboration above. This section uses the model after providing the parameters to run the optimisation. The input parameter service implements a service which provide input parameters for running evolutionary algorithms by design engineers. The study includes models with mixed objectives having both quantitative and qualitative objectives and discrete and continuous variables. The parameters provided can also be saved in the database server for future reuse using the collaboration service. The parameters and other optimisation resources are aggregated for resource sharing. As mentioned before, a MODO Aggregation (MODOA) can be defined as the incorporation of multi-objective optimisation (MOO) resources within interdependent disciplines that share these resources for mutual benefits within grid environment. These resources include data, parameter,

constraints, objective functions, computational power, data storage and expertise. The information provided by the schema gets published on the WebMDS and users can see the properties and functionalities of the parameter service.

Under Linux environment, the parameters are provided at the command prompt for NSGA-II. This research has developed a graphical user interface (GUI) that allows design experts to enter optimisation parameters conveniently. This makes the problem solving environment (PSE) user friendly and intuitive in nature. The parameter service presents 4 main interfaces for parameters to be entered. The first interface is the interface that allows design engineers to enter the population size, number of generations, number of objective functions, number of constraints and number of real variables. Figure 7.12 shows this interface.

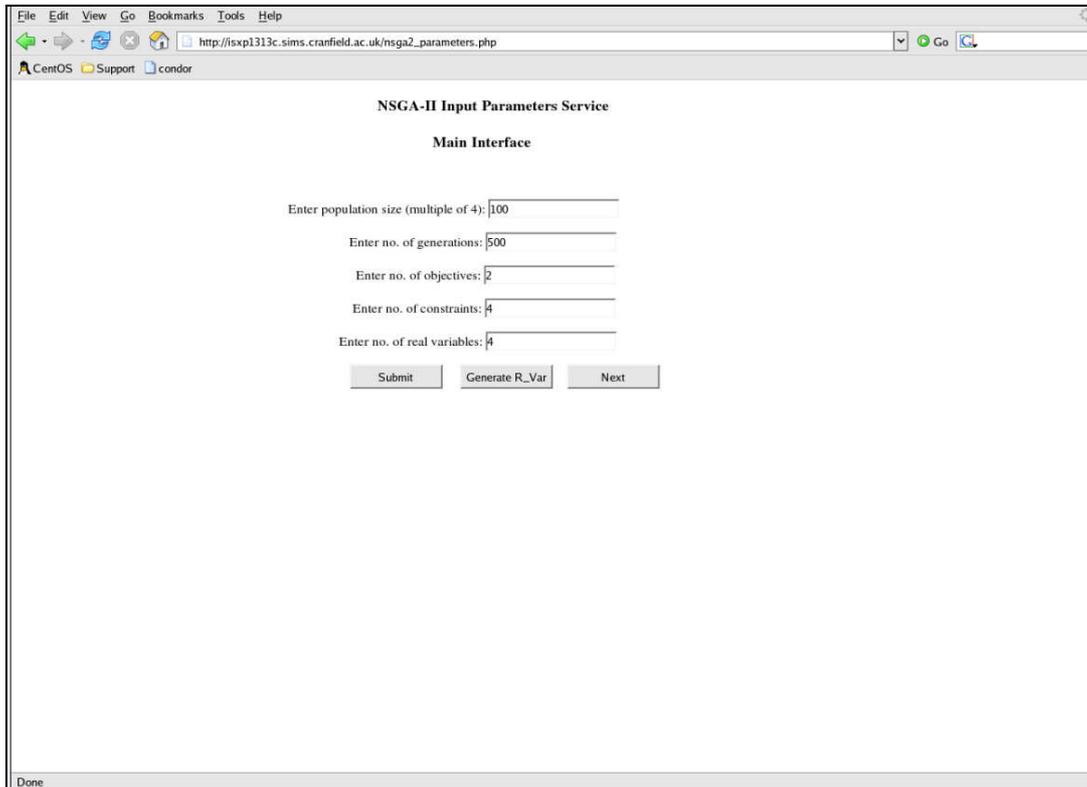


Figure 7.12: First input parameter service interface

These 5 parameters are submitted to a text file as well as to the PostgreSQL database which can be retrieved and reused when the need arises. These file and database get updated with further parameters using the append feature for files and save record capabilities in PostgreSQL. The last value (number of real variables) is used to generate a sub-interface under the first interface for the lower and upper bounds of the

real variables. The user can change the parameters in case there is a mistake. This is an improvement on conventional Linux NSGA-II input parameter feature which gets exited with any mistake without allowing the user a second chance to make changes. The lower and upper bounds restrict the optimisation process to obtain values only within these bounds.

When the lower bounds and upper bounds for the real variables have been entered, the expert is now directed to the second interface using the Next button which enables the expert to enter probability of crossover of real variable, probability of mutation of real variable, distribution crossover index, distribution mutation index and number of binary variables. In all the 3 case studies, the number of binary variables is 0 and so there is no need to discuss that interface, though it is also implemented for generality purpose. Figure 7.13 shows the second parameter input interface.

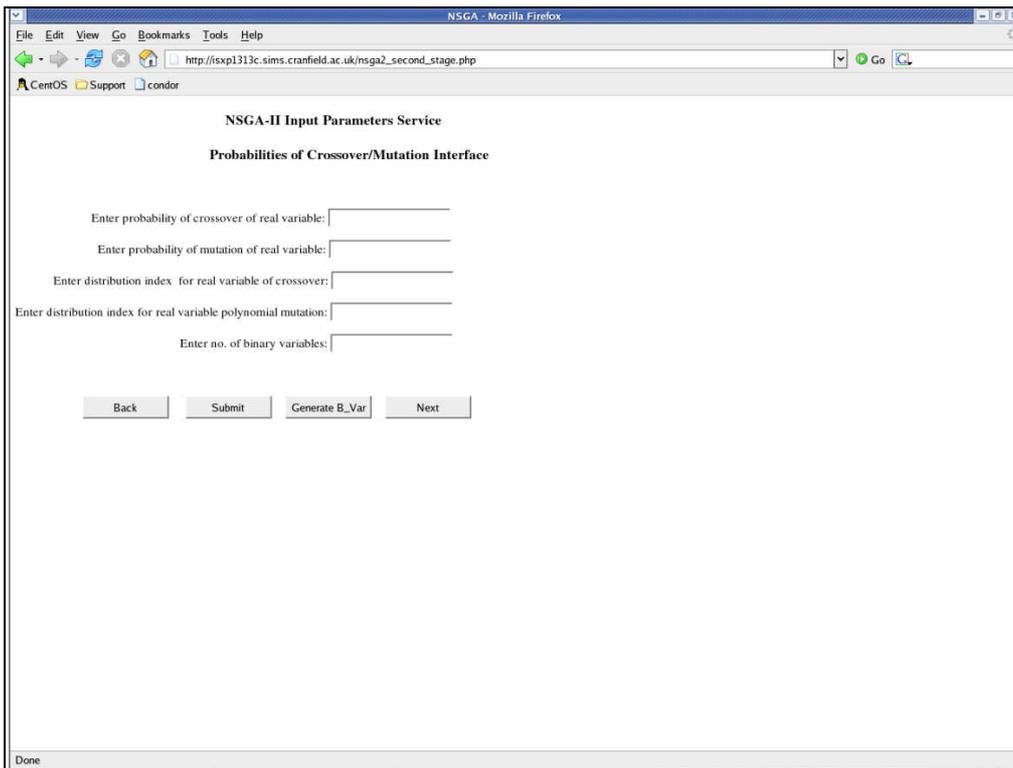


Figure 7.13: Second input parameter interface

The third interface allows the expert to enter choice of display (0 or 1), objective index for x-axis and objective index for y-axis. The choice of display means that the user may choose to display just text results (when 0 is chosen) or to display both text results and to plot the Pareto front (when 1 is chosen) using GNU PLOT. With this last

set of parameters provided, the optimisation can be run and results observed in graphical form or text format. These results will be discussed in the next chapter. Figure 7.14 is the input parameter interface for the last set of parameters required to run NSGA-II.

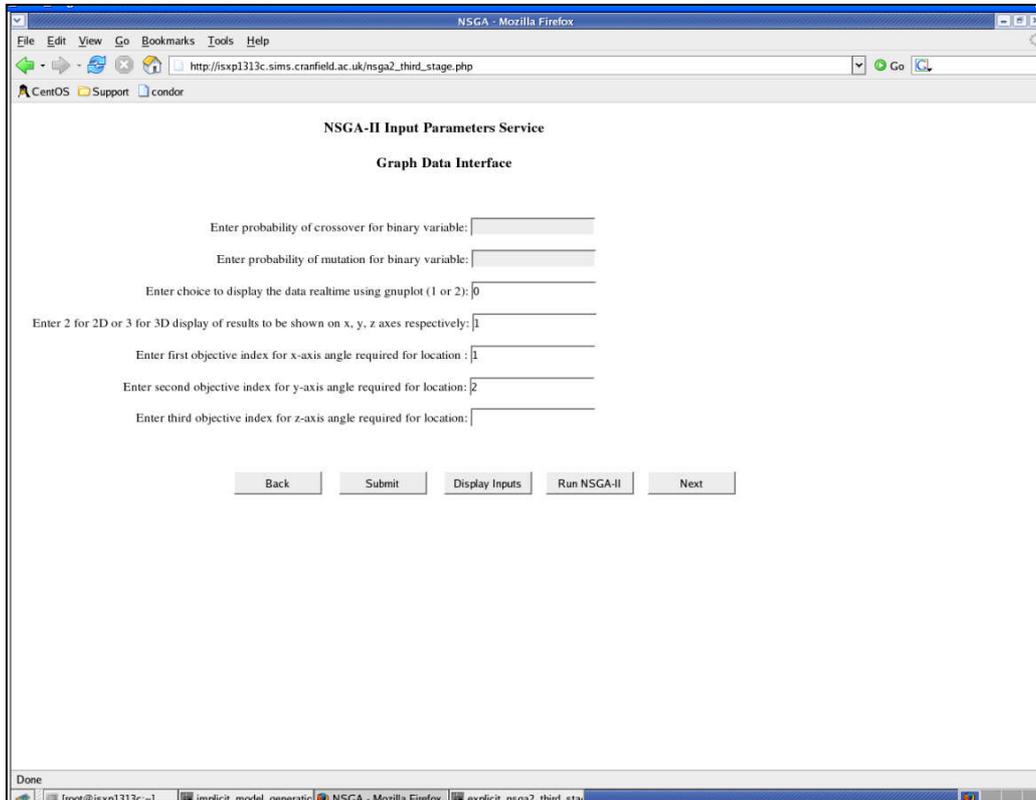


Figure 7.14: Third input parameter interface

The model built or uploaded is now linked to the problem definition file (problemdef.c) of NSGA-II. The command that is used to run the executable file is customised within the web and grid-enabled interface. At a click on the Run NSGA-II button in Figure 7.14, this command is fired and the results of the optimisation appear. The normal working principles of GAs to optimise objective functions of the model now start with the first generation of the population. The decision variables now represent the chromosome of the problem the GAs work on the population to evaluate the objective functions with respect to the constraints and decision variable bounds. The reproduction operation uses selection methods such as ranking, tournament or proportionate selection to duplicate good copies of the solution and discard bad solutions keeping the population size constant (Deb, 2001). The ranking

selection method is the one used by NSGA-II. This is because of some computational and efficient drawbacks in the tournament and proportionate selection methods (Deb, 2001).

The next operation is the crossover operation which might be one-point, two-point or more in strategy. The crossover is the operation that produces new sets of solutions. This and mutation operation will be discussed in the next chapter during validation of results using case studies. Figure 7.14 describes the process of the services.

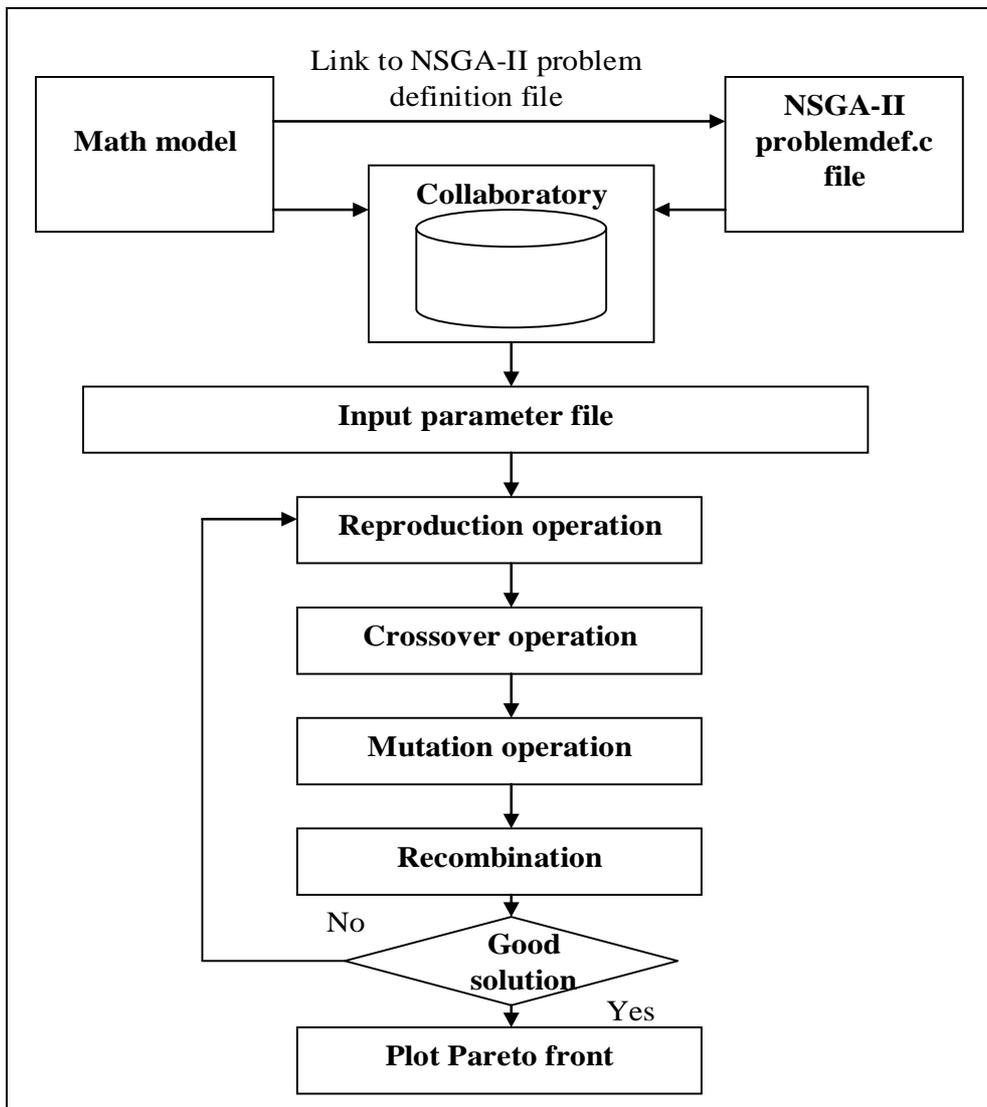


Figure 7.15: Description of math model building and parameter inputs for optimisation

7.5.6 Qualitative optimisation and visualisation services

The third case study-design of a manufacturing plant layout has two objective functions. One is quantitative and the other is a qualitative function. The quantitative

function is treated the same way that is described in section 7.5.2. However, the second objective function requires that users (experts) rate the design of the manufacturing layout from 0 to 9. This means that distributed experts will be presented with a visual display of the design and an input field or input button to enter their rating for a design. Each value of a rating represents the evaluation of the qualitative objective. The description of this problem will be discussed in detail in chapter 8. The design is implemented in C++ using Qt Design software package in the grid that runs in Linux. The visualisation service provides the animation of different designs based on the constraints which specify the maximum length and width of the entire layout and the maximum number of rooms that can be provided within the layout. Qualitative and visualisation display will be discussed in the next chapter.

7.5.7 Computational service

The optimisation processes are performed within the DECGrid which consists of 8 computers as described in section 7.3. The efficient utilisation of processing power is enhanced by allocating jobs to systems especially when they are idle. This service is provided by the Condor scheduling system. This is done by configuring the condor resource manager to allow remote nodes to submit jobs to Condor pools. A Condor pool is made up of machines that put their resources together to accomplish a computational task. Condor allows jobs to flock from one pool to another when a pool is idle. A match making algorithm is used to match job request for resources to available computational resources. Condor uses Globus GRAM service to accomplish cycle-stealing task. MODO tasks are interdependent. For example the mathematical model is built when a domain is selected and after that the criteria, parameters and constraints are provided. Condor uses Directed Acyclic Graph (DAG) concept to locate resources on the network and allocate them to jobs. DAG is a method of representing inputs, outputs or data that another process depends on as its input. The DAGMan (DAG Manager) capability in Condor is used for efficient resource workflow management. This capability is explored when inputs, outputs and some data depend on other programs. This is a common feature in MODO resource sharing.

The submission of jobs can be done by two methods. The first method is the use of Condor-G (Condor-Globus Resource Allocation Management) and the second method is using the Globus submit command. To submit a job, the submit description file is

required. This is the file which contains information on the executable file, certificate, host that the job should be sent to for execution and name of the grid. An example of a submission description file is shown Figure 7.16.

Figure 7.16 is the description file that allows users to submit multiple jobs at the same time to different machines with different architectures running different operating systems in DECGrid. This is one of the advantages of using the grid platform for job processing. It allows jobs to be parallelised into sub-jobs that are sent to different nodes for processing to speed up the computation of the job. MyProxyHost and MyProxyServerDN are variables that shows the host and server machines. Requirement is a variable that indicates the types of operating systems that this job can run on. For security reasons, password requirement usually precedes submission of jobs so that the user is verified at the host.

```
Universe           = DECGrid
Grid_resource      = gt4 condor
MyProxyHost        = isxp1314c.sims.cranfield.ac.uk:7512
MyProxyServerDN   = /O=sims.cranfield.ac.uk/OU=People/CN=Gokop Goteng
MyProxyPassword   = condorsuser
Executable         = modoFile.$$ (OpSys).$$ (Arch)
Requirements       = (OpSys == "WINDOWS" && Arch == "INTEL") ||
                    (OpSys == "LINUX" && Arch == "INTEL")
Error              = modoErrFile.err
Input              = modoInputFile.in
Output             = modoOutputFile.out
```

Figure 7.16: Condor job submission description file

The Condor configuration file is also configured in a way that jobs that are submitted from the Condor Master (server) are flocked (migrated) to worker (clients) nodes for processing especially when the processors are idle. This capability is implemented in DECGrid by configuring the FLOCK_FROM and FLOCK_TO properties as follows.

```
$FLOCK_FROM = "hostnames with semi-colon after each hostname if more than 1"
$FLOCK_TO = "hostnames with semi-colon after each hostname if more than 1"
```

\$FLOCK_FROM migrate jobs from the hosts named to the hosts listed in \$FLOCK_TO. This means that when the nodes listed in \$FLOCK_FROM have some jobs running or are occupied, they can send jobs to hosts in \$FLOCK_TO when they are idle. This ensures computational throughput of the grid. The cycle-stealing concept (using processors of systems that are idle) is implemented in Condor to speed up job processing by setting the Condor configurations in properties of StartIdleTime (amount of time for which the keyboard must be idle before starting a job at the node), KeyboardBusy (a true or false expression which indicates TRUE when the keyboard is busy and FALSE when it is not busy), CPUIdle (a Boolean expression which indicates TRUE when the CPU is idle) and CPUBusy (indicates TRUE when the CPU is busy). By configuring these properties in Condor configuration file, jobs were flocked to idle worker nodes to speed up processing and ensure efficient utilisation of the systems.

The design of DECGrid also takes into consideration the issue of resource management among distributed experts carrying out MODO. The important components for resource management are the GRAM and GASS. GRAM ensures the allocation of resources to different clients (worker nodes) for remote execution and job submission. When jobs are submitted, the request is sent to the remote host and handled by a daemon built in GRAM located in each host. The daemon creates the job manager that starts and monitors the job processing progress. The job manager notifies the client when the job has finished processing. GASS provides the capability for file transfers among the worker nodes and master server. A web-enabled GUI is used for parameter inputs that are stored in files managed by GASS. Submission of jobs using the Globus GRAM is done using the Resource Specification Language (RSF).

This research uses the concept of master-worker model to parallelise optimisation job. The optimisation algorithm is run on the master node (server) and the objective functions are parallelised to free nodes for computation. For each generation, the fitness functions are computed on separate worker nodes for each population and results sent back to the server for ranking, crossover and mutation and then back again to the worker nodes for the computation of the next round of fitness functions. This

process continues until the last generation. To do this a multiple job submission description is provided. This is shown in Figure 7.17.

Figure 7.17 enables submission of 2 jobs at the same time. The figures 1 and 2 as shown shows the number of jobs submitted. These jobs are submitted to different nodes. This is the technique used to parallelise jobs for submission in this research. The file ensures that the time of submission, execution and completion of the jobs are recorded.

```
<?xml version="1.0" encoding="UTF-8"?>
<multiJob xmlns:wsa="http://www.w3.org/addressing">
  <factoryEndpoint>
    <wsa:Address>
https://isxp1313c.sims.cranfield.ac.uk:8443/wsrf/services/ManagedJobF
actoryService
    </wsa:Address>
  </factoryEndpoint>
  <directory>/usr/local/globus-4/</directory>
  <count>1</count>

  <job>
    <factoryEndpoint>
<wsa:Address>https://isxp1314c.sims.cranfield.ac.uk:8443/wsrf/service
s/ManagedJobFactoryService</wsa:Address>
    </factoryEndpoint>
    <executable>/bin/date</executable>
      <count>2</count>
    </job>

  <job>
    <factoryEndpoint>

<wsa:Address>https://isxp1315c.sims.cranfield.ac.uk:8443/wsrf/service
s/ManagedJobFactoryService</wsa:Address>
    </factoryEndpoint>
      <count>1</count>
    </job>
</multiJob>
```

Figure 7.17: Job submission definition file

7.6 Implementation issues

In the course of implementing the DECGrid, there were issues that were observed. These issues served as learning process for the researcher. The choice of operating

system (OS) is important as one need to collaborate with other users who are familiar with the OS so that when problems arise, it is easy to put heads together and solve it. However, this negates the vision of the grid which is supposed to be an infrastructure which is platform independent. Various versions of Globus middleware do not implement certain aspects of grid services or sometimes there is the need to get some third party packages to implement some functionality. Other implementation issues are summarised below.

- Service providers strive to ensure error handling, monitor workflow progress, standardise error reports and identify dependencies and reproduction of workflows. This is not possible on many occasions if many users submit computationally and data intensive jobs.
- Service users are looking for usability features. Optimisation experts want to carry on with inputs using GUI without bothering themselves with the underlying low level implementation of the services. That is why this research has provided a user friendly GUI for parameter inputs.
- Meeting service-level agreements and guarantee for service reliability is challenging as some nodes that hold important optimisation data may fail. During this research, two nodes failed and the researcher had to reinstall the OS and middleware all over again. This is a clear case of instability of the grid infrastructure
- How can service providers validate services provided by distributed users based on service-level agreement? This question is very difficult to answer when the services required centre on qualitative rather than quantitative service level agreement. As the number of users increases, the efficiency is likely to decrease
- Building trust models to validate services provided by distributed users (based on service-level agreement) is challenging. This is why some companies do not use Globus, but built their own middleware. GRIA project is one such example.
- Some capabilities in Globus middleware are not fully supported by Windows and other operating systems, though all capabilities are supported by Linux. This may hinder collaboration with users who are using different operating systems.
- The mathematical model building service is not efficient for complex and implicit models.

7.7 Summary

This chapter described the practical process of DECGrid implementation. The facility and tools used to set up the grid infrastructure were discussed. Services such as mathematical model building service, collaboration service, parameter input and optimisation service and computational service were implemented to support MODO. Issues arising during implementation were identified to serve as lessons in future grid research implementations. The next chapter will validate the implementation with 3 case studies.

Chapter 8 - Validation using Case Studies

The previous chapter presented the implementation of DECGrid infrastructure and the services intended for use in this research for MODO applications. These services (mathematical model building, collaboration, computational and NSGA-II parameter input/optimisation services) will be used on 3 selected real-life case studies for validation. This chapter will also present and discuss the results obtained together with the methodology used for validation. The purpose of this chapter is:

- *To describe three real-life case studies*
- *To propose a validation methodology for the research*
- *To present and discuss the results obtained*
- *To validate the framework and architecture in chapter 6*
- *To validate the MODO service specifications*

8.1 Real-life case studies

This section will discuss three real-life case studies that are from MODO application areas. These case studies are taken from aerospace, manufacturing and architectural design disciplines to demonstrate their computational, data intensity, collaborative and multidisciplinary nature. Because the research involves solving multi-objective design optimisation problems, these case studies that will be discussed are MODO related problems. The criteria for selecting the 3 case studies are based on how the services designed can handle them. As mentioned above, the problems' computational intensity is handled by the computational service provided; the collaboration takes care of the collaboration and data management aspect of the problems and the mathematical model building service is used to build explicit models. Other criteria are the complexity and multi-objective nature of the problems, the types of variables (discrete or continuous), the mixture of the objectives (qualitative and quantitative or maximisation and minimisation) and the complexity of the constraints. These case studies will be discussed one after another below.

8.1.1 Aerospace: Design of a turbine blade cooling system

This case study is taken from Roy (1997) and Tiwari (2001). To enhance the performance and efficiency of gas turbine engine, the turbine blades need to operate

in an environment that the gas temperature is as high as practicable. However, this temperature sometimes exceeds the operational limits of the turbine blade material. To maintain the integrity of the blade component and at the same time operate at high gas temperatures, the blade materials are cooled to safe operating temperature levels by passing cool air on them or over them in the form of films. A small part of the compressor exit airflow is used to cool the blades. However, the temperature of this cooling air depends on the compressor pressure ratio and on the Mach number and temperature. The trade-off for the blade cooling includes loss of work and efficiency due to the portion of the air taken from the compressor exit. Hence this problem can be described as a multi-objective problem with four objectives namely coolant mass flow for radial passage (W_{cr} in Kg/s), coolant mass flow for film hole (W_{cf} in Kg/s), metal temperature for gas side (T_{wg} in K) and metal temperature for film side (T_{wf} in K). The problem has twelve variables, namely, type of geometry (Geo), coefficient of discharge (radial passage) (C_{dr}), heat transfer coefficient factor (radial passage) (F_{hc}), inlet temperature (T_{c1}), wall thickness (dth), thermal conductivity of the blade material (K_w), pressure ratio (between inlet and outlet of radial passage) ($R_p = P_{c1}/P_{c3}$), perimeter ratio (radial passage) ($R_s = S_{gr}/S_{cr}$), film hole diameter (df), coefficient of discharge (film hole) (C_{df}), heat transfer coefficient factor (film hole) (F_f) and pressure ratio (film) ($R_{pf} = (P_{c1}-P_{c2})/(P_{c1}-P_{c3})$).

The first variable (Geo) is discrete (plane, ribbed or pedestal) and the 11 other variables are real. The values of C_{dr} and F_{hc} vary within a range depending on the type of the geometry. There are 15 constraints in this problem which include blade wall temperature (for both gas and film sides) limits and flow ratio (W_{cr}/W_{cf}). There are implicit objective functions. This is a typical multi-objective optimisation problem that suits the DECGrid application platform. These multi-objective functions are implicit with outputs becoming inputs at various iterations during modelling. Figures 8.1 and 8.2 are illustrations of the arrangement of the components of the problem.

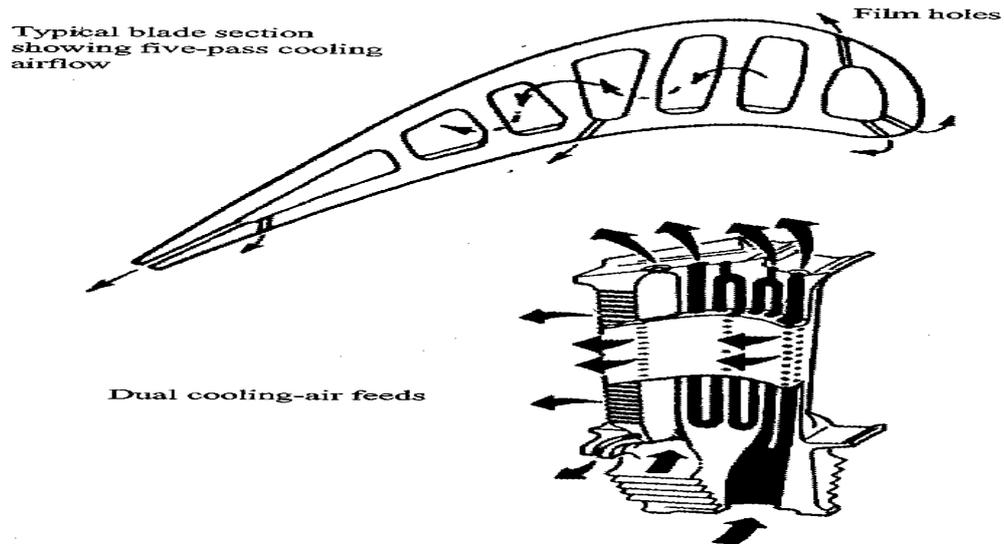


Figure 8.1: General arrangement of five-pass cooling of turbine rotor blade (Roy, 1997; Tiwari, 2001)

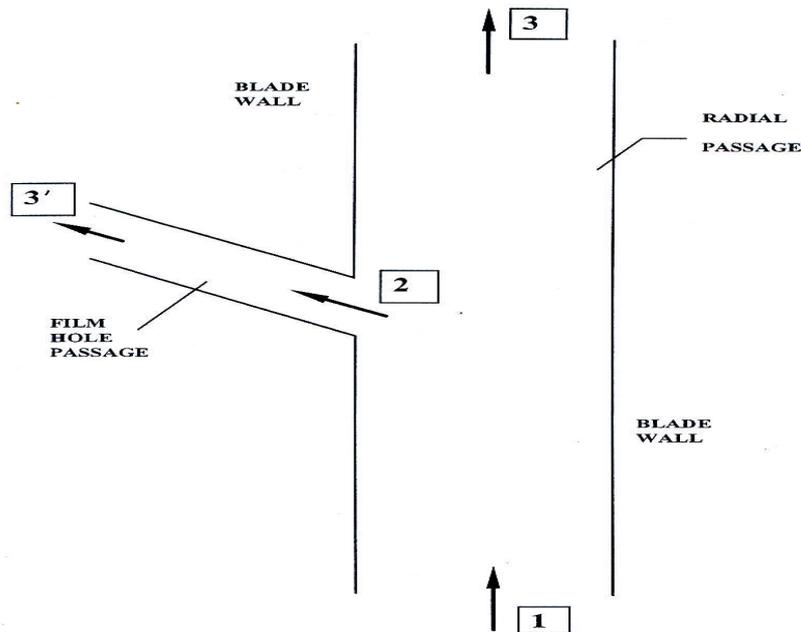


Figure 8.2: Schematic diagram showing general arrangement of coolant flow through turbine blade with film cooling mechanism (1: coolant air inlet, 2: film cooling passage inlet, 3: cooling air exit and 3': film cooling hole exit) (Roy, 1997; Tiwari, 2001)

8.1.2 Manufacturing: Welded beam problem

This case study is presented by Deb (2001). This problem describes how a beam needs to be welded on another beam and must carry a certain load. The objective of the design is to minimise the cost of fabrication and minimise the end deflection. Here, the overhang portion of the beam and the applied force (F) are specified, thus making the cross-sectional dimensions (b , t) of the beam and the welded dimensions

(h , l) as the variables. The problem has four constraints. The first constraint is to make sure that the shear stress at the support location of the beam is smaller than the allowable shear strength of the material. The second constraint is to make sure that normal stress at the support location is smaller than the allowable yield strength of the material. The third constraint is to ensure that the thickness of the beam is not smaller than the weld thickness from a practical standpoint and the fourth constraint ensures that the allowable buckling load of the beam is more than the applied load. This problem is multi-objective and has an explicit model. The problem also has a non-linear, convex and continuous non-dominated set of solutions as observed in literature. Figure 8.4 is the description of the problem.

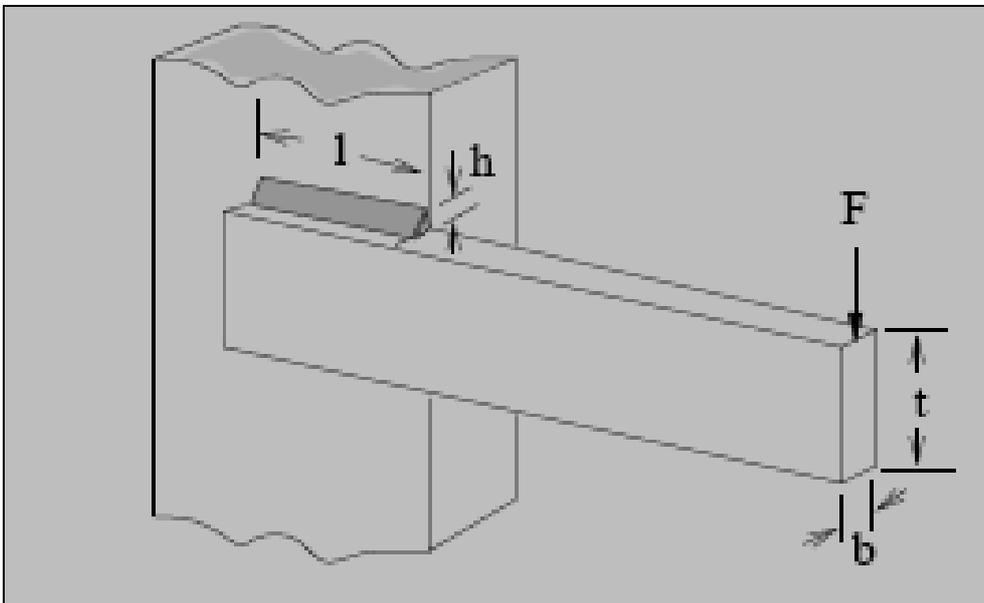


Figure 8.3: Welded beam design (Deb, 2001)

8.1.3 Architecture: Design of a manufacturing plant layout and floor planning

The design of a manufacturing plant layout is treated by Brintrup (2007) and floor planning by Sushil (1993). This is a case study with multiple (two) models that are expected to be run concurrently by two different groups of experts that must share computational and optimisation resources. This case study demonstrates the capability of the grid to provide multiple distributed design experts to work together and at the same time protect data and information that they want to preserve. This section will discuss Brintrup (2007) first and then Sushil (1993) and afterwards provide the relationship between the two. The manufacturing plant layout problem is to optimise

the arrangement of different spaces for different usage in a manufacturing plant. Manufacturing companies have different requirements and considerations for optimising the spaces available for their usage and so it is difficult to use one single method or tool for optimising a manufacturing plant for all companies. This is why the design of manufacturing plants is usually carried out using different tools, methods and involves some negotiations with experts from different fields (Brintrup, 2007). The goal of this problem is to minimise the cost of building the plant and to maximise the efficiency and satisfaction of the plant for users. The problem has two constraints. The first constraint is that the total overall width (T_w) of the plant is fixed at 3.6 units and that of the length (T_l) is 2.2 units. The problem has 8 variables which represent the widths and lengths of the various spaces. The plant must have 7 spaces namely stores, offices, assembly, warehouse, press room, paint room and spares. The cost of each space is proportional to the area of the facility, except for warehouse and press areas. All the facilities are rectangular except the assembly area. This problem has one quantitative objective and one qualitative objective. This demonstrates how different experts can qualitatively rate designs from distributed nodes as they visualise the designs. Figure 8.3 is the description of the plant layout.

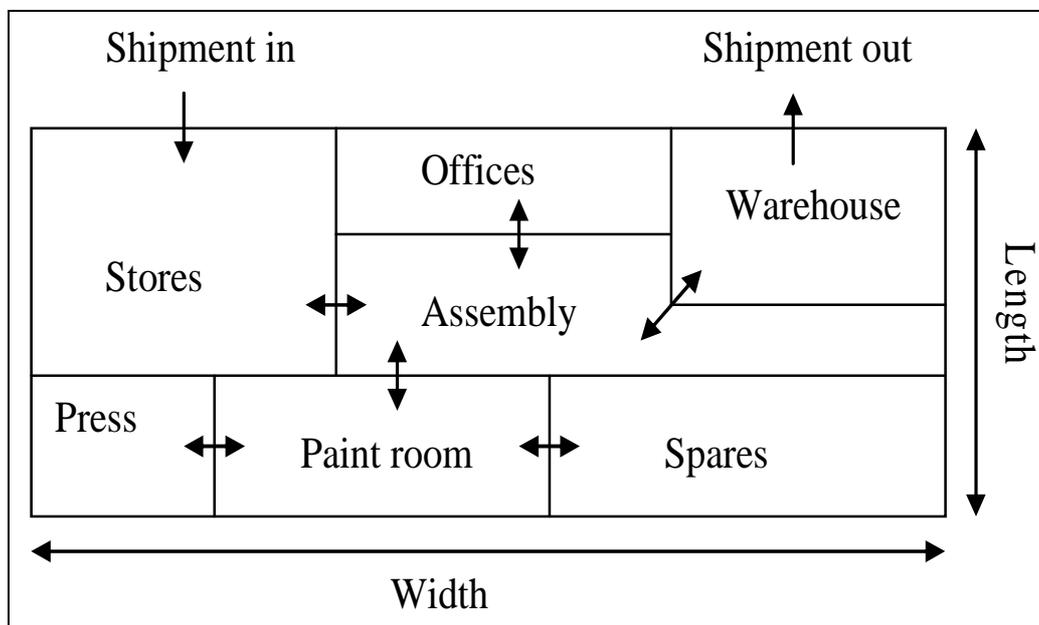


Figure 8.4: Manufacturing layout design problem (Brintrup, 2007)

The model from Sushil (1993) is to optimise the floor planning of the area where the staffers of the manufacturing plant live. This problem finds the length and width of

each room that will minimise the cost of the apartment in the presence of some constraints. Like the first model, the cost of each room is proportional to the area of the room except for the cost of kitchen and bathroom whose costs are two times their respective areas. The constraints are shown in Table 8.1.

Table 8.1: Floor planning upper/lower bounds constraints (Sushil, 1993)

Room	Length		Width		Area		Proportion
	Min	Max	Min	Max	Min	Max	
Living	8	20	8	20	120	300	1.5
Kitchen	6	18	6	18	50	120	Any
Bath	5.5	5.5	8.5	8.5			
Hall	5.5	5.5	3.5	6	19	72	Any
Bed1	10	17	10	17	100	180	1.5
Bed2	9	20	9	20	100	180	1.5
Bed3	8	18	8	18	100	180	1.5

Using these two models (Brintrup, 2007; Sushil, 1993), the researcher exploited the use of Virtual Organisation Membership Service (VOMS) in grid services to assign decentralised roles for members of the groups. VOMS has capability to manage access to computational resources between different domain users of the same grid by protecting each group’s interests as regards data ownership and autonomy of administration.

8.2 Reasons for choosing the case studies

As stated earlier, this research is using three case studies to validate the prototype designed and developed for MODO applications. The selection criteria are based on multiple objective functions, multiple constraints, mixed variable types, mixed objective functions, implicit and explicit models. Other features considered are the collaborative nature of the problem, computational and data intensity and heterogeneity of platforms requirements. The researcher decides to treat design of gas turbine blade cooling system, design of a welded beam problem and the design of a manufacturing plant layout/floor panning as the first, second, third case studies respectively. The first case study has 4 objective functions, 12 variables (1 discrete and 11 real), 3 constraints, 12 variable bounds and has implicit multi-layered

interactions in the iterations during computation. This problem is computationally intensive for large population size and generations. Each of the 4 objectives is computed at different nodes and the parameter service interface allows distributed users to enter inputs and make changes. The second case study has 2 objectives, 4 constraints, 4 variables and has non-linear convex and continuous Pareto front. The mathematical model building service is used to create this model and link it to the problem definition file in NSGA-II. The third case study has two models with each model having 2 objectives (1 quantitative and 1 qualitative). The first model (Brintrup, 2007) has 2 constraints and 8 variables that are interdependent and the second model (Sushil, 1993) has 18 constraints and 8 variables. This case study demonstrates how a quantitative objective is computed and the system has to synchronise for qualitative objective to be fed in by the experts. The distributed sources of the ratings (qualitative values) from different experts at different nodes allow for collaboration among different designers. For each model, one node was used for computation of quantitative objective and two or more nodes were used for the qualitative objective computations. This involves requirements for efficient resource sharing among collaborators.

8.3 Validation methodology

This section outlines the procedure for the validation of the case studies and prototype. It is important to determine what to validate, how to validate, who will participate in validation so that the validation can easily be linked to the contribution to knowledge. This applies to all the three case studies.

8.3.1 What to validate

The areas to validate are:

- The applicability of using service specification document for implementing services.
- The usability of DECGrid for MODO applications.
- The process of building the explicit math model.
- The math model built compared with the existing math model.
- The results produced compared to the results published in literature.

8.3.2 How to validate

There are certain steps that are used for validation. This is because the prototype is developed in the department using a dedicated room with 8 computers. This means that the researcher needed to use the researchers (experts) within the department for the validation process. Grid researchers within the university (Cranfield) were involved through consultations and discussions on the framework and architecture and were invited to the department and see how the system delivers services. Five workshops have been conducted by the researcher with the MODO experts in the department. This is also part of the in-house validation process. The following steps were used to validate each case study:

- Prepare structured questionnaire on the functionality of the DECGrid prototype for MODO applications and develop use case scenario (see section 8.3.3).
- Invite 5 experts to come to the department where the system is setup and give them a presentation and demonstration of the prototype. Section 8.3.4 describes the background of the experts that participated in the validation.
- Explain the use case scenario to the respondents.
- Issue the questionnaires to the respondents (experts) and get their feed back.
- Implement the respondents' comments.

8.3.3 What is the use case scenario

The use case scenario is the description of how users of the systems can operate the prototype. This is because the prototype is intended to serve as a problem solving environment. This means that it should be user friendly and should provide MODO experts access to tools and resources that can enhance their efficiency. The following describes one scenario on how the researchers who participated in the validation will use the system in the third case study (design of a manufacturing plant layout and floor planning):

- The optimisation algorithm runs on the master node where selection, crossover, mutation and ranking operations happen.
- A researcher (expert) works on building the quantitative (Q^T) model on one node. The interface described in Figure 7.5 is used for building the model.
- This Q^T model is then linked to the problem definition file in NSGA-II.
- Another researcher works on the qualitative (Q^L) model on another node. This means that the expert can visualise the designs, rate them and submit the ratings

(qualitative objective evaluations) to the database and file for subsequent use by NSGA-II.

- The third researcher runs the optimisation on the master node. When the Q^T and Q^L evaluations are completed, they are sent to the master node where they are merged so that NSGA-II can perform operations on them in preparation for the computation of fitness functions for the next generation.
- The fourth researcher analyses the results on yet another node. The third case study provides an interactive NSGA-II scenario. When the rating of the design is finished for a generation, the optimisation continues. This continues until the results for the last generation are computed.
- Researchers can now plot the results using GNU PLOT.

8.3.4 Who participates in validation

As mention earlier, the research needs to use the experts to validate the prototype. The researcher used the following category of experts for validation:

- Senior multi-objective design researchers in the university
- Junior multi-objective design researchers in the university

The experimental results for each case study describes the scenarios that the researcher used for the validation experts that participated in the validation and the optimisation results obtained. This separation highlights the importance of using grid to solve the case studies and conventional optimisation process. In the three case studies, 5 researchers participated in the validation. The researcher classified the experts into two namely senior and junior researchers. Senior researchers are doctoral degree (PhD) holders and junior researchers are almost completing their doctorate. Two senior researchers and three junior researchers participated. The first senior researcher had PhD in multi-objective optimisation in the laboratory that developed NSGA-II and is a Research Associate (RA) with four years experience. The second senior researcher is also a RA and has PhD in optimisation. Two of the junior researchers are doing PhD in multi-objective design optimisation and one in knowledge management.

The validation process in this research considers two major components namely validation of the grid services and validation of the results obtained compared to those

in published literature. The first aspect of the validation looks at how the server and clients communicate to accomplish parallelisation of optimisation task and how designers use the system.

The details of each case study are discussed below.

8.4 Design of gas turbine blade cooling system

This case study has been introduced in section 8.1.1. Figures 8.1 and 8.2 described the interacting components that make up the Gas Turbine Blade Cooling System Model (GTBCSM). The twelve variables and four objective functions for the problem have also been described earlier. GTBCSM problem also have some constants known as design parameters as well as three non-linear constraints.

This section will describe the mathematical model in detail and obtain an algorithm for writing the programming codes that will be implemented in NSGA-II. There are common symbols that are used in the GTBCSM and will be described in alphabetical order in Table 8.2.

Table 8.2: Common symbols in GTBCSM

Symbol	Description
A	Cross-sectional area of passage
C_d	Coefficient of discharge
C_p	Specific heat at constant pressure
C_v	Specific heat at constant volume
D	Hydraulic diameter
D_{th}	Wall thickness
H	Heat transfer coefficient
H1	Parameter group for heat balance equation
H2	Parameter group for heat balance equation
H3	Parameter group for heat balance equation
K	Thermal conductivity
L	Passage length
M	Mach number
N	Number of (e.g. N _b stands for number of blades)
P_c	Cooling air pressure
R	Gas constant
S_c	Cooling side perimeter
S_g	Gas side effective perimeter
T_c	Cooling air temperature
W	Mass flow
X_f	Distance from film cooling hole exit/effect slot width of film
Γ	Ratio of specific heats

μ	Dynamic viscosity
-------	-------------------

Some subscripts that are used in the model are 1 for cooling air inlet, 2 for film cooling passage inlet, 3 for cooling air exit, 3' for film cooling hole exit, b for blade, c for coolant, f for film, g for gas, hpc for high pressure compressor, r for radial passage and w for wall. Table 8.3 describes the constants or design parameters used in GTBCSM.

Table 8.3: Constants or design parameters for GTBCSM

Design parameter	Description
$h_g=3000.0 \text{ W/m}^2\text{K}$	Heat transfer coefficient (gas side)
$T_g=1500.0 \text{ K}$	Gas side temperature
$\Gamma=1.36$	Ratio of specific heats
$W_{hpc}=84.85 \text{ Kg/s}$	Mass flow (high pressure compressor)
$P_{c3}=460000.0 \text{ N/m}^2$	Radial cooling hole exit pressure
$N_b=78$	Number of blades
$T_{wg}=1250.0 \text{ K}$	Wall temperature (gas side) for initial calculations
$l_r=0.0406 \text{ m}$	Radial passage length
$C_p=993.0$	Specific heat at constant pressure
$F=0.01855$	One of 2 factors for heat transfer coefficient
$R=287.0$	Gas constant
$X_f=10$	Distance from film cooling hole exit/effective slot width of film
$Mach=0.6$	Mach number
$N_f=30$	Number of film holes
$T_{wg1}=1500.0 \text{ K}$	Initial outside temperature
$A_{cr}\leq 2.75E-05 \text{ m}^2$	Maximum radial passage area
$100.0 \text{ W/m}^2\text{K} < h_{cr} < 4000.0 \text{ W/m}^2\text{K}$	Bounds on radial coolant flow heat transfer coefficient
$1000.0 \text{ K} < T_{wg} < 1500.0 \text{ K}$	Check on metal temperature
$h_f=h_g$	Heat transfer coefficients are the same for the film and gas side for film cooling section
$R_{sf}=1.0$	Perimeter ratio for film cooling section

Now that the design variables and design parameters for GTBCSM have been identified, the procedure for iterative computation for modelling the objective functions will be illustrated. As mentioned during the selection criteria of case studies in section 8.2, this problem has implicit objective functions that are multi-layered. To compute the objective functions, an iterative process that is computationally intensive is required. The computational service in DECGrid which uses GRAM and Condor is applied here to flock the computations to idle nodes to speed it up. The mathematical formulation is described below.

$$W_{cr} = 0.003 \times W_{hpc} / N_b \quad \text{Equation 8.1}$$

$$h_{cr} = h_g (S_{gr} / S_{cr}) \frac{(T_g - T_{wg})}{(T_{wc} - T_c)} \quad \text{Equation 8.2}$$

$$FF = F \times Fhc, \quad \text{Equation 8.3}$$

$$k = \frac{2.978E-03 \times T_c^{0.5}}{1 + (240.0 / T_c)}, \quad \text{Equation 8.4}$$

$$\mu = \frac{1.488E-06 \times T_c^{1.5}}{T_c + 110.4} \quad \text{Equation 8.5}$$

(The assumption for initial value is that $T_c = T_{c1}$)

$$A_{cr} = (FF \times (k / \mu^{0.8})(W_{cr}^{0.8} / h_{cr})), \text{ where,} \quad \text{Equation 8.6}$$

$$W_{cr} = \frac{A_{cr} C_d P_{c1}}{\sqrt{T_{c1}}} \left(\frac{2\gamma}{R(\gamma-1)} \left(\left(\frac{P_{c1}}{P_{c2}} \right)^{-2/\gamma} - \left(\frac{P_{c1}}{P_{c2}} \right)^{-(1+\gamma)/\gamma} \right) \right)^{0.5} \quad \text{Equation 8.7}$$

$$h_{cr} = FF \times (k / \mu^{0.8}) \times (W_{cr}^{0.8} / A_{cr}^{0.9}) \quad \text{Equation 8.8}$$

$$T_{wg} = \frac{\left(1 + H2 - \frac{H1 \times H2}{H1 + H3} \right) T_g + \left(H1 - \frac{H1^2}{H1 + H3} \right) T_{c1}}{1 + H2 - \frac{H1 \times H2}{H1 + H3} + \frac{H1 \times H3}{H1 + H3}}, \text{ where,} \quad \text{Equation 8.9}$$

$$H1 = \frac{h_{cr}}{h_g \times (S_{gr} / S_{cr})}, \quad \text{Equation 8.10}$$

$$H2 = (h_{cr} S_{cr} l_r) / (2W_{cr} C_p), \quad \text{Equation 8.11}$$

$$S_{cr} = 3.545 \times \sqrt{A_{cr}}, \quad \text{Equation 8.12}$$

$$l_r = 0.0406 m,$$

$$H3 = \frac{0.5k_w}{dth \times h_g} \left(1 + \frac{1}{(S_{gr} / S_{cr})} \right), \quad \text{Equation 8.13}$$

$$T_c = (H2 / H1)(T_g - T_{wg}) + T_{c1}, \text{ where,} \quad \text{Equation 8.14}$$

$$T_{c3} - T_{c1} = 2(T_c - T_{c1})$$

The complexity of the model as outlined above requires a step by step breakdown on how to go about the task of coding the problem. Table 8.4 is an illustration of these steps.

Table 8.4: GTBCSM programming procedure (Sources: Roy, 1997; Tiwari, 2001)

Step	Task	Equation	Description
1	Estimate W_{cr}	Equation 8.1	Use limiting value of flow off-take from the engine compressor
2	Estimate T_{wg}	-	Use material property limitation, suggested 1500.0 K
3	Calculate h_{cr}	Equation 8.2	-
4	Calculate A_{cr}	Equation 8.6	First confirm the value, if within the limiting value of A_{cr} , go to step 5, else set $W_{cr}=W_{cr} \times 0.99$ and go back to step 4.
5	Calculate W_{cr}	Equation 8.7	-
6	Calculate h_{cr}	Equation 8.8	Compare h_{cr} value from step 6 with the value in step 3. If within tolerance then proceed to confirm whether h_{cr} lies within the acceptable limit and if yes then proceed to step 7 else reset the values of T_{wg} and h_{cr} and go back to step 4. If the wall temperature calculation reaches a steady state then only accept, if not equal then go back to step 4.
7	Calculate T_{wg}	Equation 8.9	Check the value of T_{wg} , if within the acceptable range then accept. If not and W_{cr} has not been changed previously, then change W_{cr} as $W_{cr}=W_{cr} \times 1.01$
8	Calculate T_c	Equation 8.14	-
9	Recalculate k	k defined in Equation 8.4	-
10	Recalculate μ	μ defined in Equation 8.5	Reset T_{wg} and h_{cr} values and go to step 4. Only accept if the wall temperature calculation reaches a steady state

The above model described only two objectives (W_{cr} and T_{wg}). These objectives are for the gas side. This is called bi-objective optimisation since the geometry (Geom) is limited and here only two objectives are considered. The model can incorporate additional two objectives to take care of the film cooling system to achieve a more effective cooling mechanism in the turbine blade. This means that additional equations for W_{cf} and T_{wf} need to be included using similar procedures as above. To begin this, it is assumed that the coolant temperature, T_c obtained above will be used as a starting point for the film mechanism, thus:

$$T_{c2} = T_c$$

And the total film cross sectional area is obtained as:

$$A_f = N_f \frac{1}{4} \pi (df)^2 \quad \text{Equation 8.15}$$

The pressure ratio across the film hole is given by:

$$\frac{Pc_2}{Pc_3} = \frac{Pc_2}{Pc_3} = \frac{Pc_1}{Pc_3} \frac{Pc_1}{Pc_2} \quad \text{Equation 8.16}$$

$$W_{cf} = \frac{A_f C_{df} Pc_2}{\sqrt{Tc_2}} \left(\frac{2\gamma}{R(\gamma-1)} \left[\left[\frac{Pc_2}{Pc_3} \right]^{\frac{-2}{\gamma}} - \left[\frac{Pc_2}{Pc_3} \right]^{\frac{-(1+\gamma)}{\gamma}} \right] \right)^{0.5} \quad \text{Equation 8.17}$$

Cooling side heat transfer coefficient h_{cf} is given as:

$$h_{cf} = FF' \times \left(\frac{k}{\mu^{0.8}} \right) \left(\frac{w_{cf}^{0.8}}{A_f} \right) \quad \text{Equation 8.18}$$

$$FF' = Ff \times F$$

The blade wall temperature downstream of the film temperature, T_{wf} is given by:

$$T_{wf} = \frac{\left(1 + H2 - \frac{H1 \times H2}{H1 + H3} \right) T_f + \left(H1 - \frac{H1^2}{H1 + H3} \right) T_{c1}}{1 + H2 - \frac{H1 \times H2}{H1 + H3} + \frac{H1 \times H3}{H1 + H3}} \quad \text{Equation 8.19}$$

Where,

$$T_f = T_g - \varepsilon_f (T_g - Tc_3) \quad \text{Equation 8.20}$$

$$\varepsilon_f = 0.66 - 00.0092 \times \left(RWA \times \frac{A_{cf} C_{df}}{W_{cf}} \times X_f \right)^{0.8} \times \left(\frac{Tg}{Tc_3} \right)^{0.6} \quad \text{Equation 8.21}$$

$$RWA = Mach \times Pc_3 \left(\frac{\gamma}{R \times t_g} \right)^{0.5} \quad \text{Equation 8.22}$$

Where,

$$t_g = \frac{T_g}{\left[1.0 + \frac{\gamma-1}{2.0} \times Mach^2 \right]}$$

$$H1 = \frac{h_{cr}}{h_g}$$

$$H2 = \frac{h_{cf} S_{cf} l_f}{2W_{cf} C_p}$$

$$S_{cf} = N_f \times \pi \times df$$

$$l_f = 5.0 \times df$$

$$dthf = dth / 2.0$$

$$H3 = \frac{k_w}{dth \times h_g}$$

As has been discussed earlier, $S_{gf}/S_{cf}=1.0$ and other relations for S_{cr} and l_r (radial passage length) have been obtained (Roy, 1997). More relationships can be obtained from Roy (1997). There are 3 constraints that ensure that the design model does not cross material and flow limits. The constraints are given in Table 8.4 but will be repeated here for clarity. They are:

- The blade wall temperature on the gas side, $1200.0 \text{ K} < T_{wg} < 1300.0 \text{ K}$
- The blade wall temperature on the film side, $T_{wf} < 1300.0 \text{ K}$ and
- The flow ratio, $W_{cr}/W_{cf} \geq 0.8$

8.4.1 Experimental results

In the turbine blade cooling system, the researcher aims to establish collaboration between MODO experts and the benefits of using grid services. The five researchers assumed the roles of resource owners on five nodes of DECGrid. By resource owners, it means resources on those nodes belong to them and they control users' access to the resources. The server runs the algorithm (NSGA-II) as a service which can be used by the distributed experts. One of the experts uploaded the model using implicit model interface described in chapter 7 and linked it to the NSGA-II model definition file. The job is submitted for optimisation. The four objective functions are parallelised to the four nodes of the other four experts for computation using the processing powers of the nodes. This speeds up the computation than using single machine. The five experts viewed the results of the optimisation from the distributed nodes and each make comments using the observation and response interface (see Appendix-III) provided by collaboration service. The experts were given sets of different parameters to use and check the results given to them by the researcher from literature. The distributed experts enter the parameters using the parameter input interface to run the

optimisation for some couple of times. This continued until the results matched or looked better than the results in literature.

The benefits of this scenario are to demonstrate:

- Computational resource sharing among MODO experts.
- Collaboration in decision making.
- Secure data transfer of optimisation resources using the grid file transfer protocol.
- Optimum utilisation of computational resources.

The experimental simulation runs described two sets of results. The first set considered only two objectives (W_{cr} and T_{wg}) for the optimisation. The result of this set is shown in Figure 8.5.

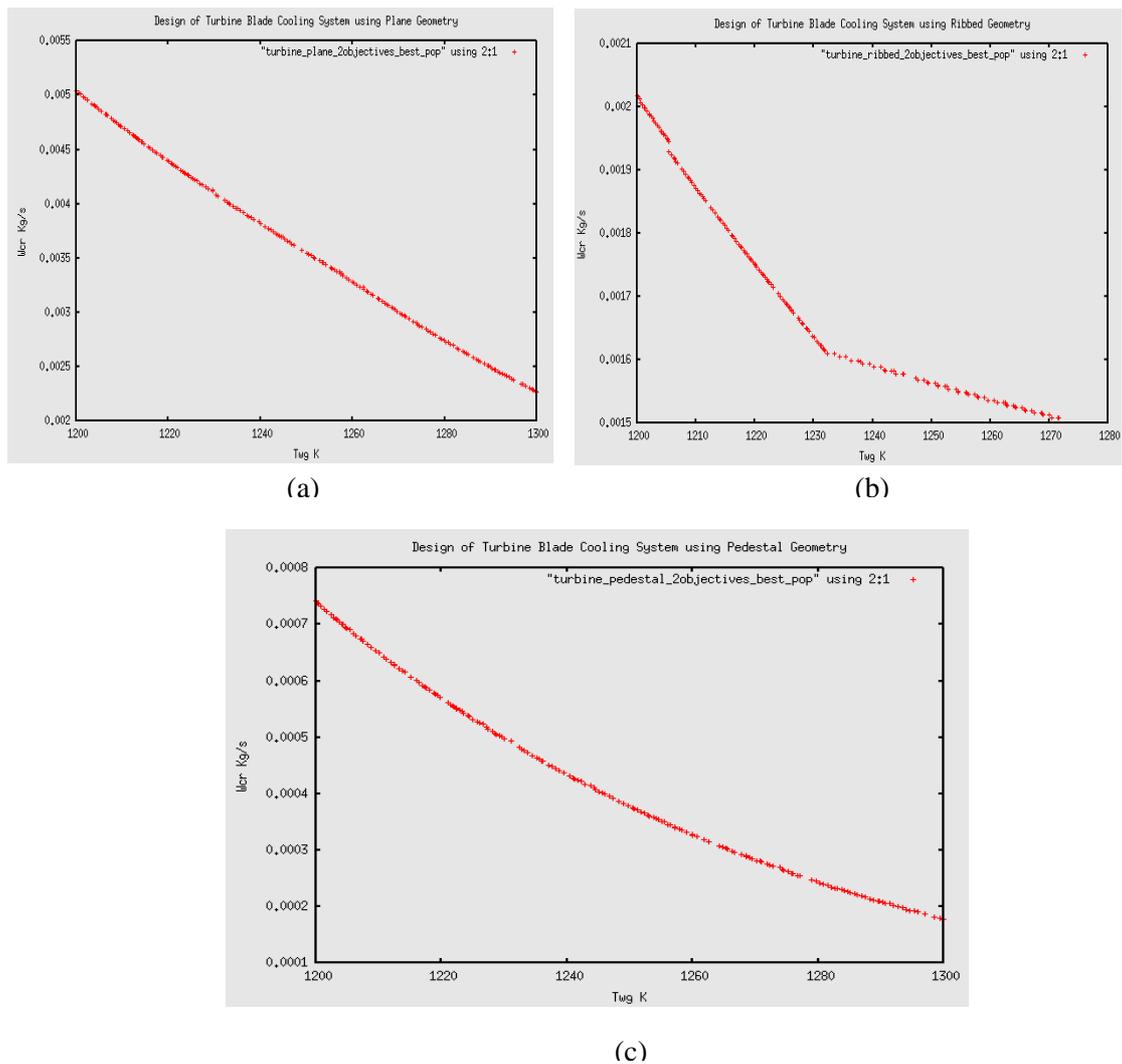


Figure 8.5: Results from NSGA-II using two objectives [Plane =(a), Ribbed=(b) and Pedestal=(c)]

Figure 8.5 shows the optimisation results using the 3 types of geometry separately to indicate the difference in material used and also to show how the discrete variable (Geom) affects the optimisation results even for two objectives. Figures 8.5 (a), 8.5 (b) and 8.5 (c) show the results of using plane, ribbed and pedestal geometries respectively. In both sets (2 and 4 objectives), the experiment is carried out using a population size of 2000 for 5000 generations, 0.8 crossover probability, 0.05 mutation probability, 10 crossover distribution index, 50 mutation distribution index and a seed value of 0.65. The boundary conditions specified in Table 8.6 were used for ribbed geometry. For the plane and pedestal geometries, the C_{dr} and F_{hc} were replaced by $\text{Plane}[C_{dr}(0.60, 0.75), F_{hc}(1.0, 1.6)]$ and $\text{Pedestal}[C_{dr}(0.20, 0.40), F_{hc}(1.80, 3.20)]$ (Roy, 1997) respectively with all other boundary conditions remaining unchanged. The results show that pedestal requires the least coolant mass while plane geometry requires the highest coolant mass to keep the blade temperature at good condition. This also highlights the fact that the amount of coolant required depends on the quality of the material used. In all the 3 geometries, there is an increase in temperature as the coolant mass decreases and vice versa.

Interesting results were also obtained by using all the four objectives for the 3 geometries. Figures 8.6, 8.7 and 8.8 show the results for plane, ribbed and pedestal geometries for the four objectives using the same boundary conditions as in the case of using two objectives.

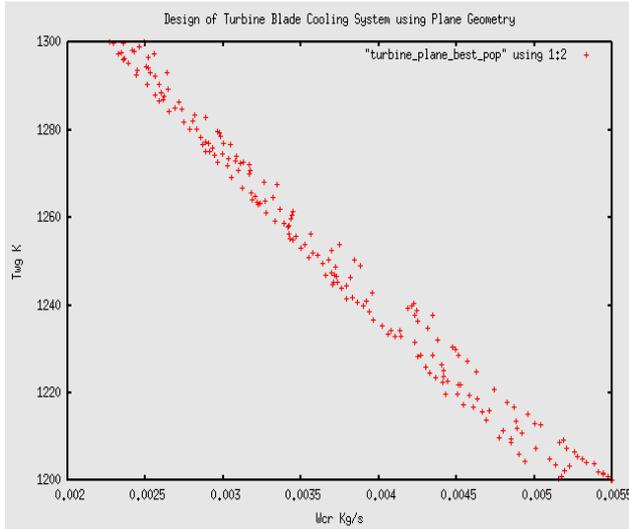
In the two sets in Figures 8.5, 8.6, 8.7 and 8.8 it can be observed that Pareto fronts are obtained in all the plots that involve W (W_{cr} or W_{cf}) and T (T_{wg} or T_{wf}). This means that there is conflict between W and T . The Pareto fronts show the trade-offs based on this conflict. This is expected in any case from the definition of the problem in section 8.1.1, meaning any increase in coolant mass flow is expected to lower the metal temperature and vice versa. Expectedly also, this conflicting behaviour is not witnessed in any of W - W (W_{cr}/W_{cf} and W_{cf}/W_{cr}) and T - T (T_{wg}/T_{wf} and T_{wf}/T_{wg}) plots.

The search space shows bias in the plots. For example in Figures 8.6 (a), 8.7 (a) and 8.8 (a), there is a bias towards higher values of T_{wg} . These plots also show that the model has a local and global Pareto front with regards to W_{cr} and T_{wg} . This multimodal behaviour arises due to the discrete variable (Geometry) in the problem.

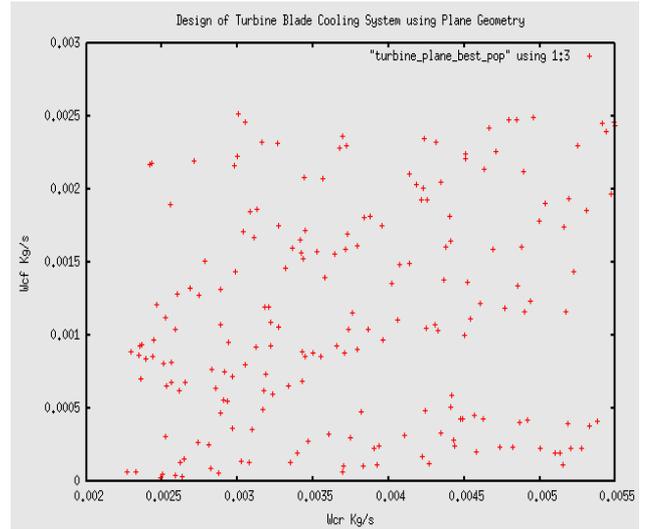
This again causes low density of population in the region between the two fronts, which may lead to deception in the search space. The results obtained from the four objectives show that the pedestal geometry requires the least coolant mass to cool the blade while the plane geometry requires the most coolant mass. The ribbed geometry indicates the combination of both plane and pedestal as it has the layers that appear in the two.

It is important to describe how the architecture in chapter 6 is used to run the optimisation problems. This section will use the parallelization concept to describe how fitness functions were parallelized to ensure computational throughput using the computational service. The server-client architecture is described here as the master-worker where the master node (server) runs the NSGA-II and also hosts the file that stores the fitness function values. The worker nodes receive fitness functions from the master and perform the computation using the population. The results obtained are sent back to the master where reproduction, crossover, mutation and ranking are performed for the first generation. This process is repeated for subsequent generations. Figure 8.5 shows the master-worker arrangement. One of the components for resource control and administration used in this research in the master-worker concept is the Virtual Organisation Membership Service (VOMS) which allows systems administrators to assign different roles and access rights to users. This is done by keeping user roles in a database and manipulating these roles based on which user should have access to what data. It also allows users to be members of multiple virtual organisations with multiple roles. These roles have expiration times to ensure security of the system.

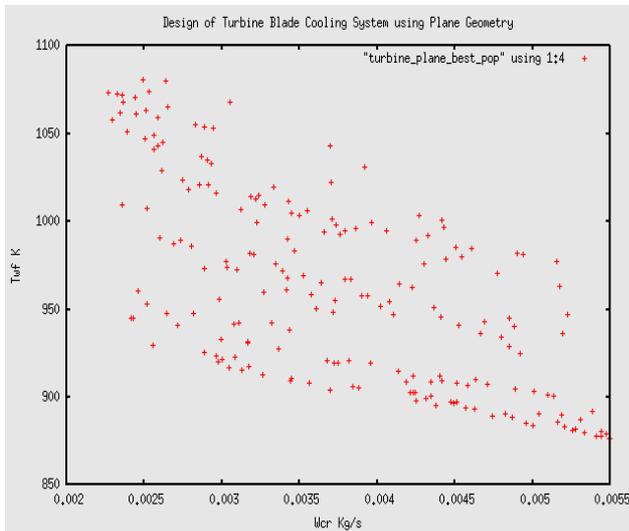
On the master side, the job description file contains the instructions to parallelise the computation of objective functions and assign each to a free worker node. The designer first checks the properties of each worker node on the web service group to ensure that complex fitness functions are assigned to nodes with higher processing capacity. This is a manual approach. An automated approach is done by configuring the Condor scheduling system to assign jobs with higher computational requirements to nodes with higher processing capacity. The transfer of data between workers and master is enhanced using the GridFTP.



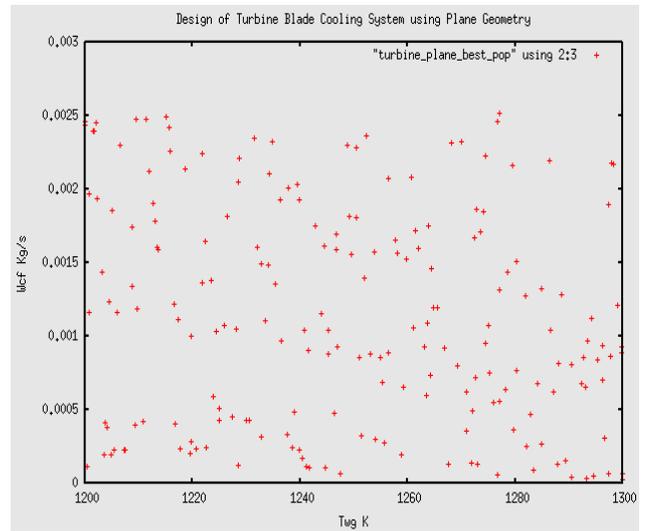
(a)



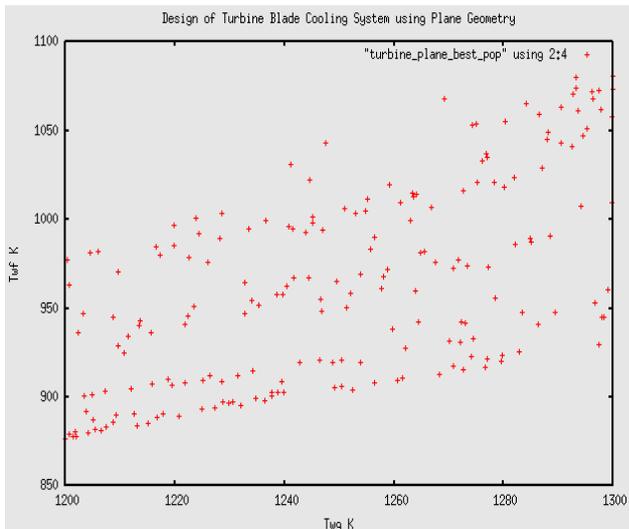
(b)



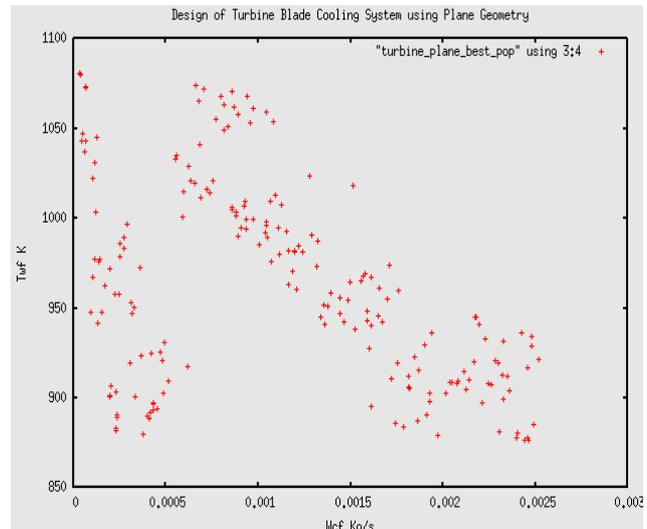
(c)



(d)

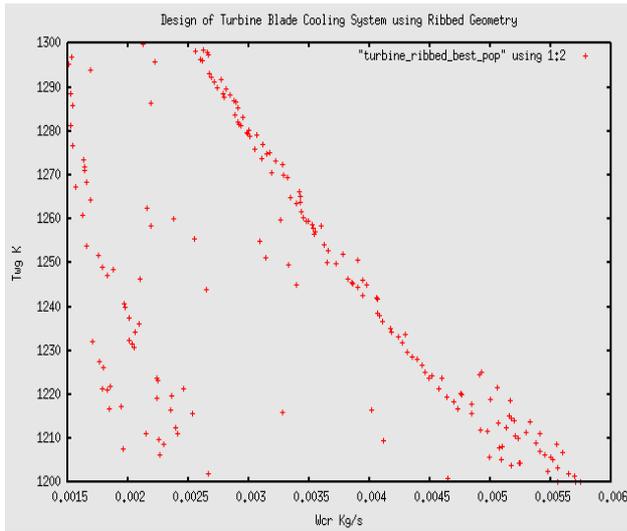


(e)

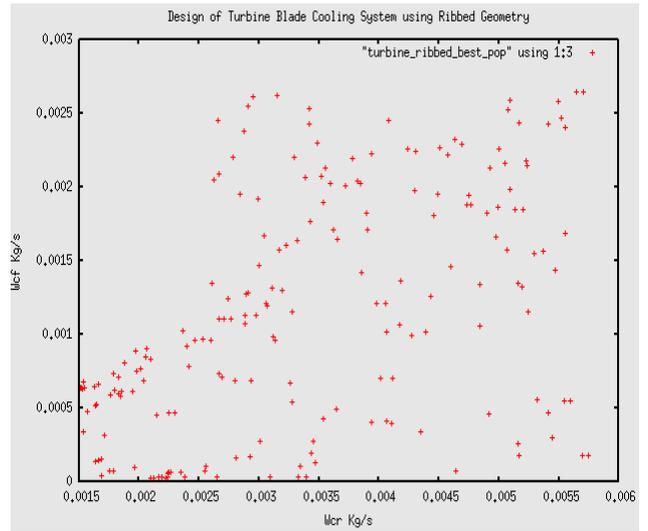


(f)

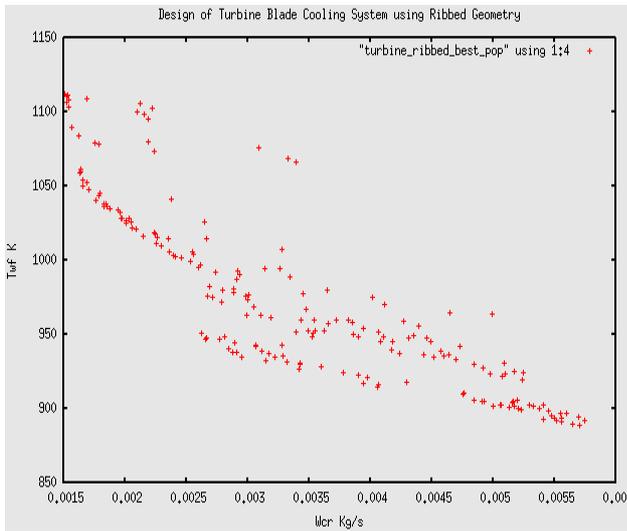
Figure 8.6: Results from NSGA-II using four objectives for plane geometry



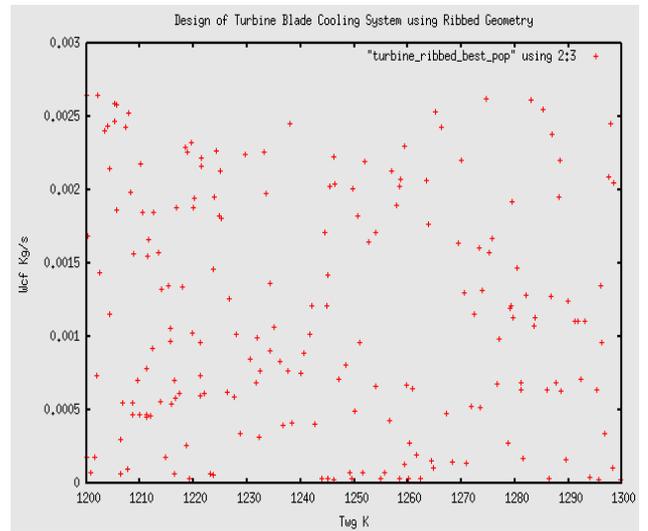
(a)



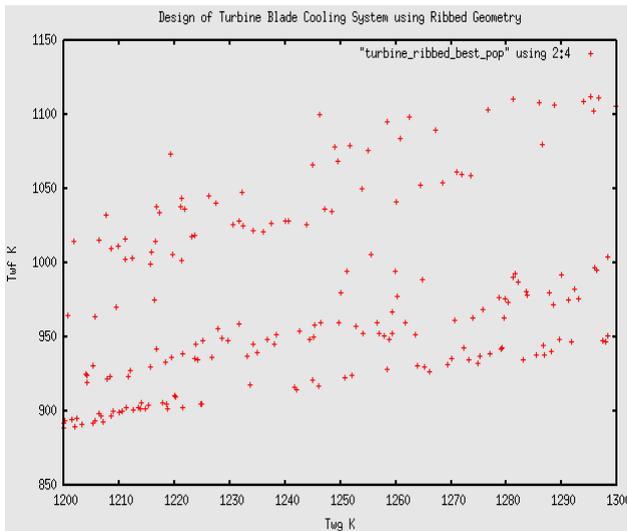
(b)



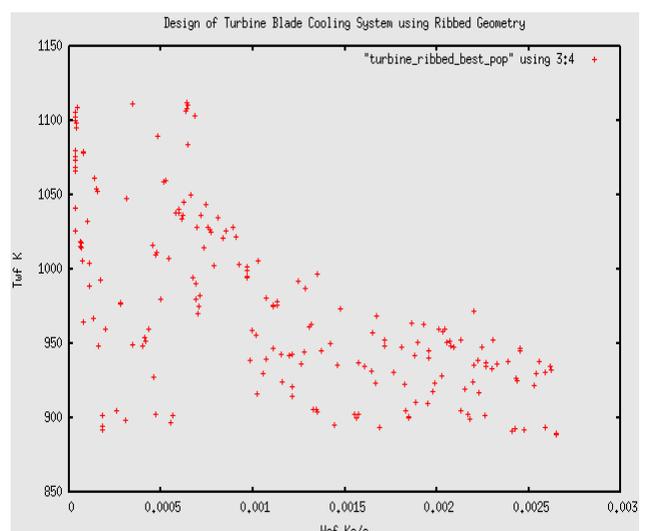
(c)



(d)

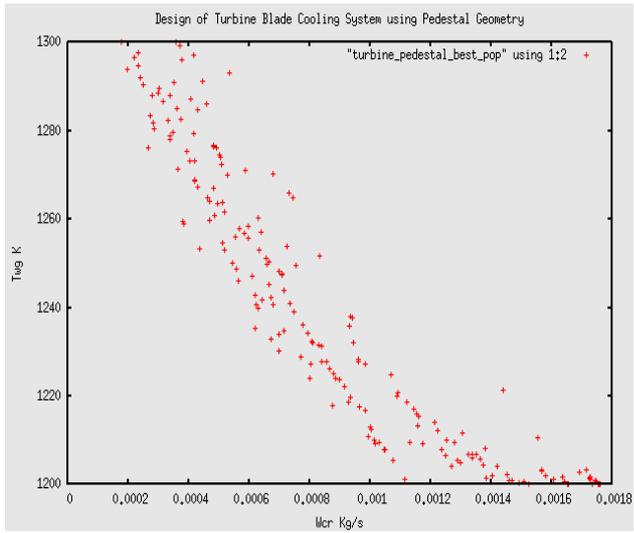


(e)

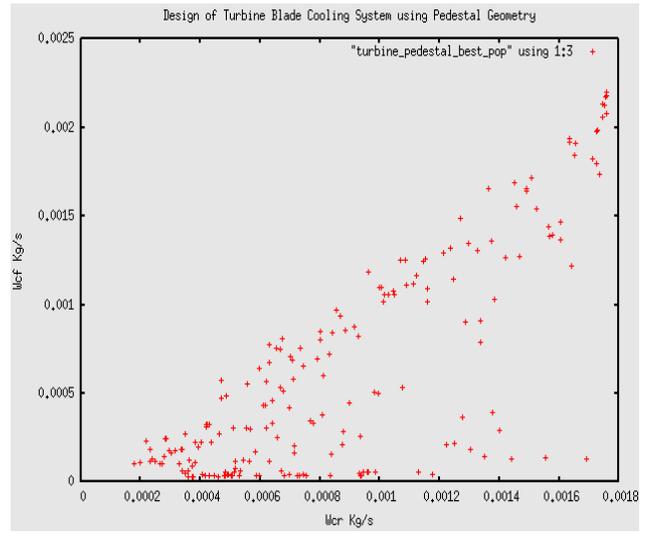


(f)

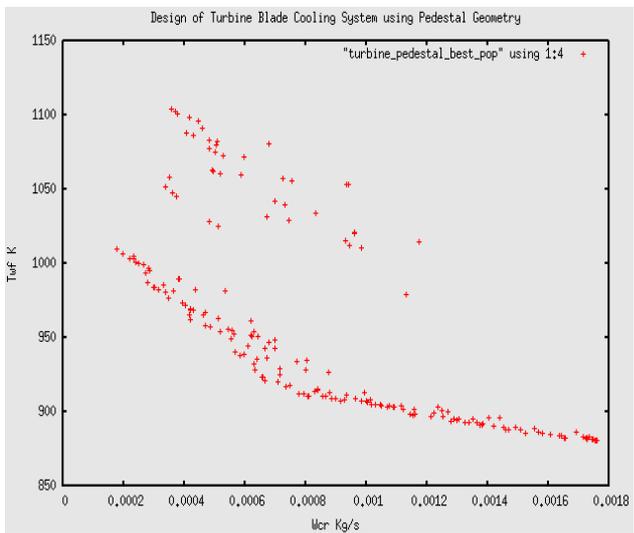
Figure 8.7: Results from NSGA-II using four objectives for ribbed geometry



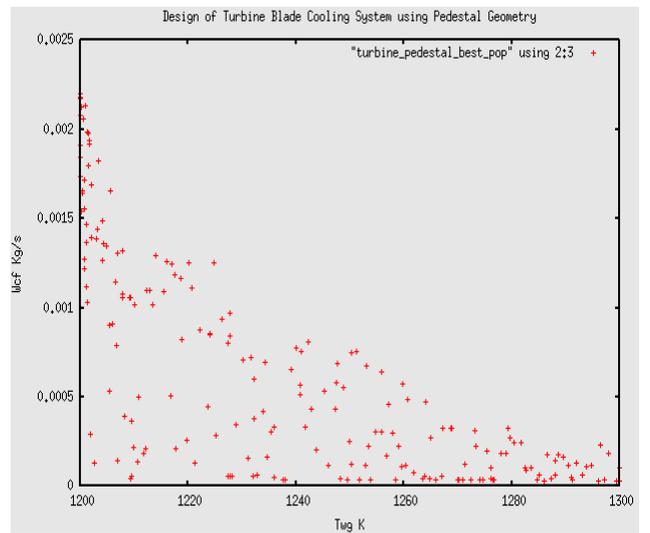
(a)



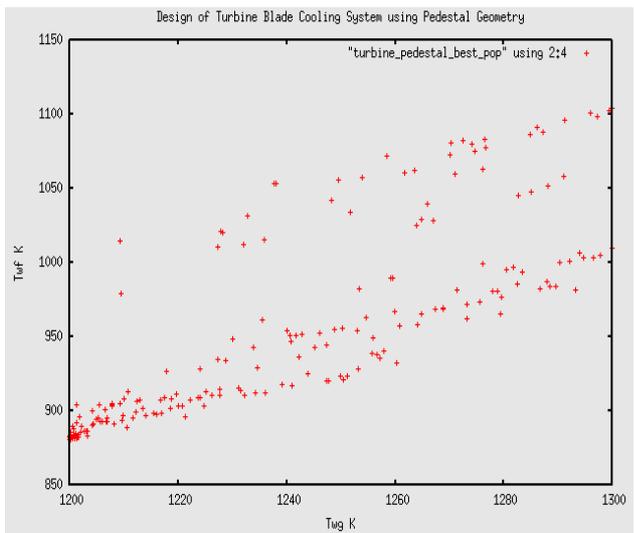
(b)



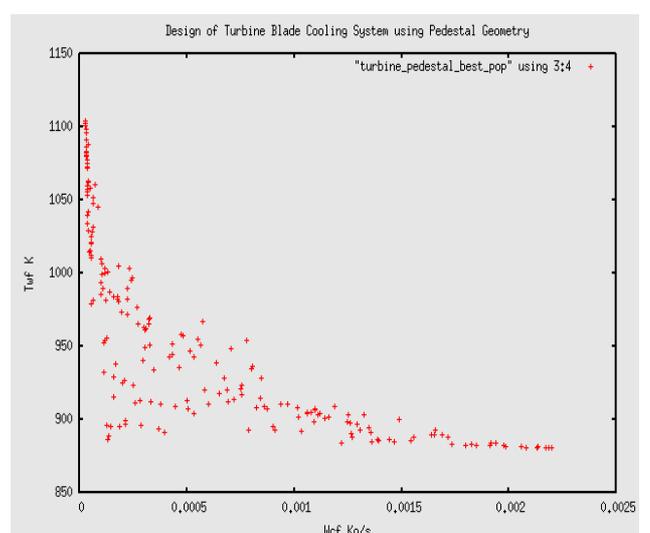
(c)



(d)



(e)



(f)

Figure 8.8: Results from NSGA-II using four objectives for pedestal geometry

Table 8.5: Variable bounds used to obtain the results in Figures 8.5

S/N	Variable	Lower bound	Upper bound
1	Geom	1	3
2	C _{dr}	0.4	0.6
3	F _{hc}	1.3	3.0
4	T _{c1}	700.0	800.0
5	Dth	0.00075	0.00250
6	k _w	18.0	33.0
7	R _p	1.05	1.60
8	R _s	0.5	1.5
9	Df	0.00010	0.00040
10	C _{df}	0.60	0.75
11	F _f	1.0	1.6
12	R _{pf}	0.20	0.40

The GRAM allows the remote execution of the job description file and in conjunction with Condor enables the user to submit multiple jobs at a time. The parallelisation of the fitness functions is done by subdividing the model into modules. All the variables are written in one module and the parameters are also coded in a separate module. These two modules are declared public so that the module for each objective function can use the variables and parameters. This avoids duplication of parameters such as population size, number of generations and variable boundary conditions for every objective function. Each objective function is also written as a module. This allows for easy parallelisation and assignment of each objective function to particular worker nodes for processing.

To submit a job execution file, the GRAM call is initiated for authentication of the user using the SimpleCA component of GSI and a synchronisation of the jobs processing task at different nodes is handled by the Condor scheduling system. This synchronisation incurs some overhead. This overhead is more when using ordinary FTP (File Transfer Protocol) protocol for file migration between master node and worker nodes. The synchronisation is achieved by simply writing a script that writes the values of fitness functions to a file when the process status of the job indicates 'complete' and waits if there is any objective function(s) whose status is idle. When all the status of objective functions indicates complete, their values that have been written in a file are sent back to the master node where reproduction, crossover, mutation and ranking are performed by the NSGA-II. The process is repeated until the

last generation. The details of latency effect of the synchronisation process will be discussed later.

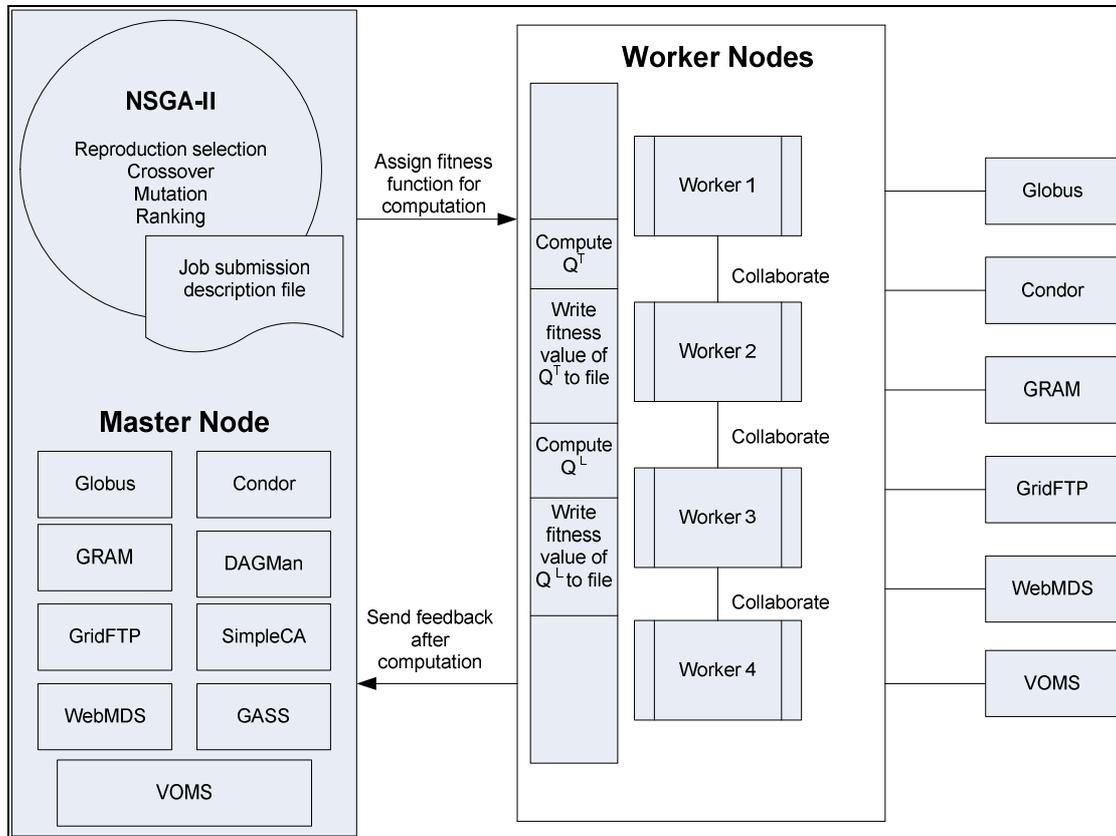


Figure 8.9: Master-worker representation

Using the implementation of master-worker, the fitness functions for the turbine blade cooling are parallelised and sent to different free worker nodes for computation. This speeds up the process as can be seen in Figure 8.19. The mathematical model is uploaded and transferred to where the NSGA-II definition is located in the master node using GridFTP which is faster than the FTP. The parameter input interface provides a convenient means for designers to enter parameters and make changes easily. This interface also guides the users to follow sequential steps for carrying out the optimisation process.

8.5 Design of welded beam problem

This case study was discussed in section 8.1.3. Figure 8.4 is the description of the problem. The problem as said earlier is to minimise the cost of fabrication given as

$f_1(x)$ and to minimise the end deflection given by $f_2(x)$. The model is given in Equation 8.23.

$$\text{Variables} = x = (b, t, h, l)$$

$$\text{Cost} = f_1(x) = 1.10471 \times h^2 \times l + 0.04811 \times t \times b \times (14.0 + l) \quad \text{Equation 8.23}$$

$$\text{Deflection} = f_2(x) = \delta(x) = 2.1952 / t^3 \times b \quad \text{Equation 8.24}$$

Subject to constrains:

$$g_1(x) = 13600 - \tau(x) \geq 0 \quad \text{Equation 8.25}$$

$$g_2(x) = 3000 - \sigma(x) \geq 0 \quad \text{Equation 8.26}$$

$$g_3(x) = b - h \geq 0 \quad \text{Equation 8.27}$$

$$g_4(x) = P_c(x) - 6000 \geq 0 \quad \text{Equation 8.28}$$

Where,

$$\tau(x) = \sqrt{\tau'^2 + \tau''^2 + (l \times \tau' \tau'') / \sqrt{0.25 \times \{l^2 + [h + t]^2\}}}$$

$$\tau' = 6000 / \sqrt{2 \times h \times l}$$

$$\tau'' = \frac{6000 \times (14 + 0.5 \times l) \times \sqrt{0.25 \times \{l^2 + [h + t]^2\}}}{2 \times \{0.707 \times h \times l \times [l^2 / 12 + 0.25 \times (h + t)^2]\}}$$

$$\text{Stress} = \sigma(x) = 504000 / t^2 \times b$$

$$\text{Pressure} = P_c(x) = 64746.022 \times (1 - 0.0282346 \times t) \times t \times b^3$$

This model assumes the following parameters:

Overhang portion of the beam = 355.6 mm

F=6000kN

Allowable shear strength of the material =97.8MPa

Allowable yield strength of the material = 206.8MPa

8.5.1 Experimental results

For the welded beam problem, the researcher demonstrated how to build a mathematical model using the mathematical model building service. The problem is first divided into tasks that can be performed by distributed users (researchers). The objective $f_1(x)$ is supplied from one node by one researcher, $f_2(x)$ is built on another node by the second researcher and the first two constraints are produced from yet another different node. The last two constraints are built by the fourth researcher. The parameter and bounds of the problem are supplied by another researcher using the parameter input interface service. Each researcher first selected the domain (welded beam problem) and used separate node for this. The researchers used the explicit mathematical model building interface to create the relations. They followed the step-by-step process as provided by the criteria (outputs) definition, parameter (inputs) definition and constraints definition interfaces. The fifth researcher used the web service group to view the available optimisation algorithm. Search strategy selection interface (see section 7.5.3) is used to search for the algorithm. The researcher then links the model to the NSGA-II definition file and runs the algorithm from the server. Again the objective functions are parallelised and submitted to different nodes for computation. The parameter input interface is used to enter parameters such as population size, number of generations and so on. The result of the optimisation is displayed on each researcher's machine.

The benefits of this scenario are to:

- Demonstrate the capability of mathematical model building and optimisation services.
- Demonstrate resource sharing and collaboration.
- Flexible search strategy selection.

This scenario also helped in validating the first two objectives of the research. This means that it has proved that the requirements for grid environment for MODO are used to develop functional service specifications for MODO. The description is below. Figure 8.10 is the result obtained from the optimisation process using NSGA-II optimisation service. Table 8.6 shows the values used as parameters for the optimisation. For example the population size used is 100 and number of generations

is 500. 20 runs are tested with different seed values used for the random number generator.

As can be seen in Figure 8.10, the Pareto front is convex and continuous. The use of different worker nodes to build the model by different experts and still maintaining autonomy of the part of the model built demonstrates the use of grid for designers to collaborate and work together. The two objective functions are submitted for computation to different nodes to speed up the computation.

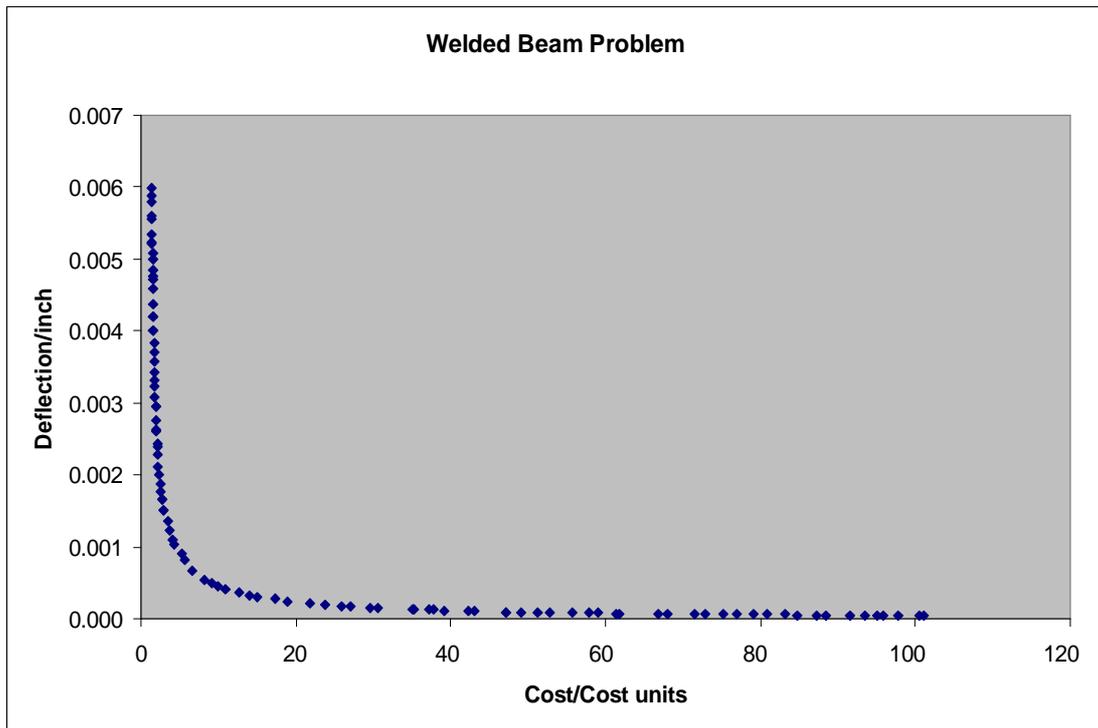


Figure 8.10: Results of optimisation of welded beam problem using NSGA-II

Table 8.6: Welded beam input parameters

Parameter	Description
100	number of generations
500	number of population
2	number of objectives
4	number of constraints
4	number of real variables
2	lower bound for first variable
10	upper bound for first variable
0	lower bound for second variable
15	upper bound for second variable
1	lower bound for third variable
3	upper bound for third variable

1	lower bound for third variable
5	upper bound for fourth variable
0.8	probability of crossover of real variable
0.05	probability of mutation of real variable
10	Distribution index for real variable crossover
50	Distribution index for real variable polynomial mutation
0	number of binary variables
1	enter 1 to display gnuplot or 0 to display only results
1	x axis objective index
2	y axis objective index

8.6 Design of a manufacturing plant layout and floor planning

The design of the manufacturing plant and floor planning were briefly introduced in section 8.1.2. Again the discussion will first start with manufacturing plant layout. From Figure 8.3, it can be seen that the lengths of the press area, paint room and press area are kept the same. Without this condition, the problem will have 14 variables instead of the 8 mentioned earlier. The quantitative objective function is given as:

$$f_1(x, y) = \sum_{i=1}^7 C_i \quad \text{Equation 8.29}$$

Subject to the following two constraints:

$$g_1(x, y) = 3.6 - \sum_0^7 W_i \quad \text{Equation 8.30}$$

$$g_2(x, y) = 2.2 - \sum_0^7 L_i \quad \text{Equation 8.31}$$

Where f_i is the function that computes the total cost of the building and C_i represents the cost functions for each parameter (width or length) of the 7 facilities as mentioned in section 8.1.2. These parameters are different for every design. This is because the cost of the facility is proportional to the area (width x length) of the facility and as the cost is computed for each generation, the values keep changing and so is the design. The parameters are the real variables that constitute the chromosome. W_i is the parameter widths and L_i is the parameter lengths of the manufacturing layout. The

qualitative values are obtained from users who evaluate the designs based on criteria given to them. For example, the researchers are told that the store and office must be bigger because they are the most favoured areas. This now forms the criteria through which the users rate the designs. There is need for consistency in this so that a realistic Pareto front can be obtained.

$$f2(x,y) = \text{Qualitative ratings given by the designer} \quad \text{Equation 8.32}$$

Figure 8.11 is the implementation interface that is used by users to visualise and rate the designs for each generation.

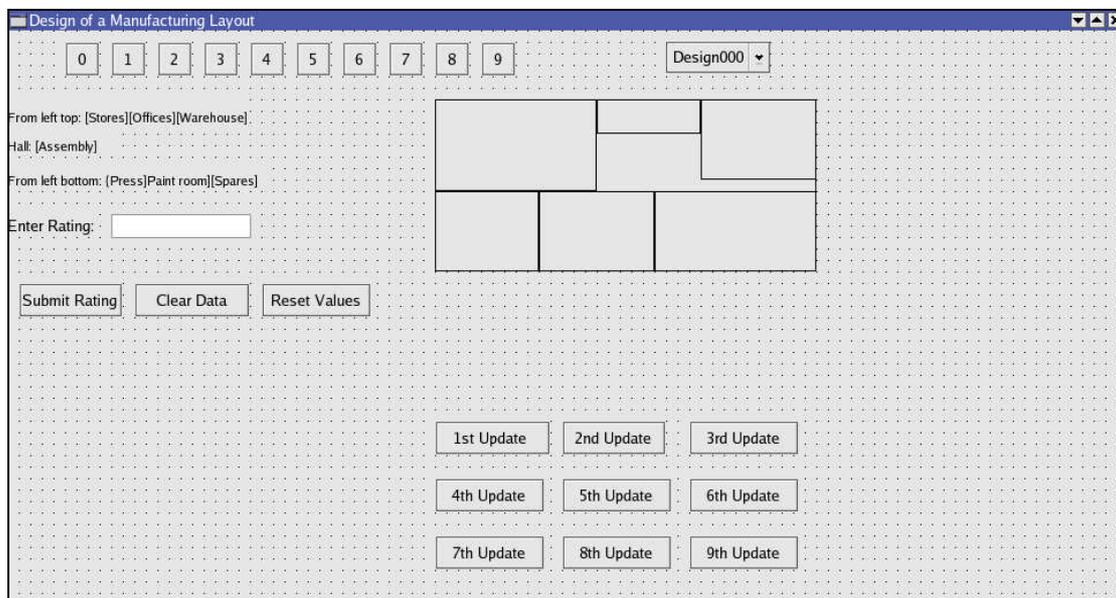


Figure 8.11: Implementation interface for rating the designs. (At the top left, when the buttons 0 to 9 are clicked, the values are saved in a file as the objective evaluations; at the top right, the designs are chosen to show new animations of each design)

The design problem parameters for the quantitative objective are shown in Table 8.7.

Table 8.7: Quantitative problem parameters for the manufacturing layout design (Brintrup, 2007).

Room	Parameter	Parameter variable	Cost function
Stores	Width	X_0	$C_1 = X_0(2.2 - X_1)$
Press area	Length	X_1	
Press area	Width	X_2	$C_2 = 2X_1X_2$
Paint room	Width	X_3	
Paint room	Length	X_4	$C_3 = X_3X_4$
Offices	Length	X_5	$C_4 = 2[3.6 - (X_0 + X_6)] X_5$
Warehouse	Width	X_6	

Warehouse	Length	X_7	$C_5 = X_6 X_7$
Spares area	-	-	$C_6 = X_1 [3.6 - (X_2 + X_3)]$
Assembly area	-	-	$C_7 = [3.6 - (X_0 + X_6)] [2.2 - (X_1 + X_5)] + X_6 [2.2 - (X_1 + X_7)]$
First constraint	Sum of parameter widths	g_1	$g_1 = [3.6 - (X_0 + X_2 + X_3 + X_6)]$
Second constraint	Sum of parameter lengths	g_2	$g_2 = [2.2 - (X_1 + X_4 + X_5 + X_7)]$

The second model is also described here. This problem is nonlinear and the lengths represent horizontal dimensions and widths the vertical dimensions. The different variables are shown in Table 8.7.

Table 8.8: Variable dimensions for floor planning model

Room	Dimension	Variable
Living	Length	Y_1
Living	Width	Y_2
Kitchen	Length	Y_3
Kitchen	Width	Y_4
Bed1 (first bed room)	Length	Y_5
Bed2 (second bed room)	Length	Y_6
Bed2 (second bed room)	Width	Y_7
Bed3 (third bed room)	Width	Y_8

Just as in the first model, there appear to be 14 variables but on a close analysis, the variables are reduced to 8. This is because some of the variables can be calculated by formulating mathematical relations from the other variables. For example the width of the hall is equal to the width of the living room minus the width of the bathroom.

The quantitative objective function that minimises the cost of the floor planning problem is given as:

$$f_1(y, z) = Y_1 Y_2 + 2Y_3 Y_4 + 93.5(Y_3 - 8.5) + 5.5(Y_2 - 8.5) + Y_4 Y_5 + Y_6 Y_7 + Y_6 Y_8$$

Subject to the constraints:

Room	Constraint	Equation
	$Y_1, Y_2 \leq 20$	
	$Y_1, Y_2 \geq 8$	

Living	$Y_1 Y_2 \leq 300$	Equation 8.33
	$Y_1 Y_2 \geq 120$	Equation 8.34
	$Y_1 / Y_2 \leq 1.5$	Equation 8.35

Kitchen	$Y_3, Y_4 \leq 18$	
	$Y_1, Y_2 \geq 6$	
	$Y_1 Y_2 \leq 120$	Equation 8.36
	$Y_1 Y_2 \geq 50$	Equation 8.37

Hall	$5.5(Y_2 - 8.5) \leq 72$	Equation 8.38
	$5.5(Y_2 - 8.5) \geq 19$	Equation 8.39

Bed1	$Y_5, Y_4 \leq 17$	
	$Y_5, Y_4 \geq 10$	
	$Y_5 Y_4 \leq 180$	Equation 8.40
	$Y_5 Y_4 \geq 100$	Equation 8.41
	$Y_5 / Y_4 \leq 1.5$	Equation 8.42

Bed2	$Y_6, Y_7 \leq 20$	
	$Y_6, Y_7 \geq 9$	
	$Y_6 Y_7 \leq 180$	Equation 8.43
	$Y_6 Y_7 \geq 100$	Equation 8.44
	$Y_6 / Y_7 \leq 1.5$	Equation 8.45

Bed3	$Y_8, Y_6 \leq 18$	
	$Y_8, Y_6 \geq 8$	
	$Y_8 Y_6 \leq 180$	Equation 8.46
	$Y_8 Y_6 \geq 100$	Equation 8.47
	$Y_6 / Y_8 \leq 1.5$	Equation 8.48

Doorways in hall	$Y_7 - 8.5 \geq 3$	Equation 8.49
	$Y_8 - Y_4 \geq 3$	Equation 8.50

Because the plan is rectangular therefore:

$$Y_1 + 5.5 = Y_3 + Y_5$$

$$8.5 + (Y_2 - 8.5) + Y_4 = Y_7 + Y_8 = Y_2 + Y_4$$

In the same manner as the first model, users are asked to rate the designs using a scale of 0 to 9 which represents the values of the qualitative objective function. The qualitative objective function is given as:

$$f_2(y, z) = \text{Qualitative ratings given by the designer} \quad \text{Equation 8.51}$$

Where f_2 is the qualitative function.

The two models have distributed users physically providing values for the qualitative fitness functions as they rate designs while still maintaining autonomy over the administration of their data using the GRAM feature and VOMS facility. This case study aims at demonstrating how grid is used to enable MODO experts to run multiple models. The NSGA-II algorithm is run on the master node and quantitative fitness function submitted to a worker node for computation. The designs are produced from another node and the user rates the designs. The same action can be performed by different groups on different models. This ensures efficient utilisation of computational resources.

8.6.1 Experimental results

This section discusses the experimental results obtained for the design of manufacturing plant layout and floor planning. The users again are the 5 researchers described earlier. The users were asked to sit on each of the nodes. The experiment was run for 10 generations for a population size of 12. Each user was facing the back of another and they are not allowed to communicate. Each generation consisted of 12 designs corresponding to the size of the population. The default designs were generated from the random initial parameter values. The users were asked to start rating the designs at the same time. The rating criteria for the first model as stated earlier was based on how big the store and office are because they are favoured while for the second model favours the living rooms. After each set of designs, each user generated the next set of designs by clicking the buttons that generate the designs. For example, after rating the first 12 (Design000 to Design011) default designs, the user will click the 1st Update button in Figure 8.11 and Design012 to Design023 (designs 13 to 24) will be generated. The users again rate these newly generated designs. Each qualitative rating has a corresponding quantitative fitness evaluation. For each generation, the quantitative values wait for the qualitative values to be entered by the users and then the NSGA-II will carry out the reproduction, crossover and mutation operations to produce the next set of solutions. It is based on these next set of solutions that the next set of designs are generated. At the end of the process, the average quantitative objective values and average qualitative objective values for each generation are obtained. Figures 8.12, 8.13 and 8.14 show the results obtained from the plots for the first model (manufacturing plant layout).

From the results in Figure 8.12, it can be seen that the average quantitative values tend to converge with successive runs. This means that the cost of building the manufacturing plant decreases with successive generations during the optimisation process. This is in line with the objective of minimising the cost of building the plant. The average qualitative values in Figure 8.13 increase with successive number of generations. That means that the designs improve in quality and the experts rate them higher with successive generations. This is in line with the maximisation of this objective. This shows that there is some consistency in the way the users have rated the designs, though the qualitative fitness function is showing some irregularities at some points. This cannot be ruled out as there is human factor in this.

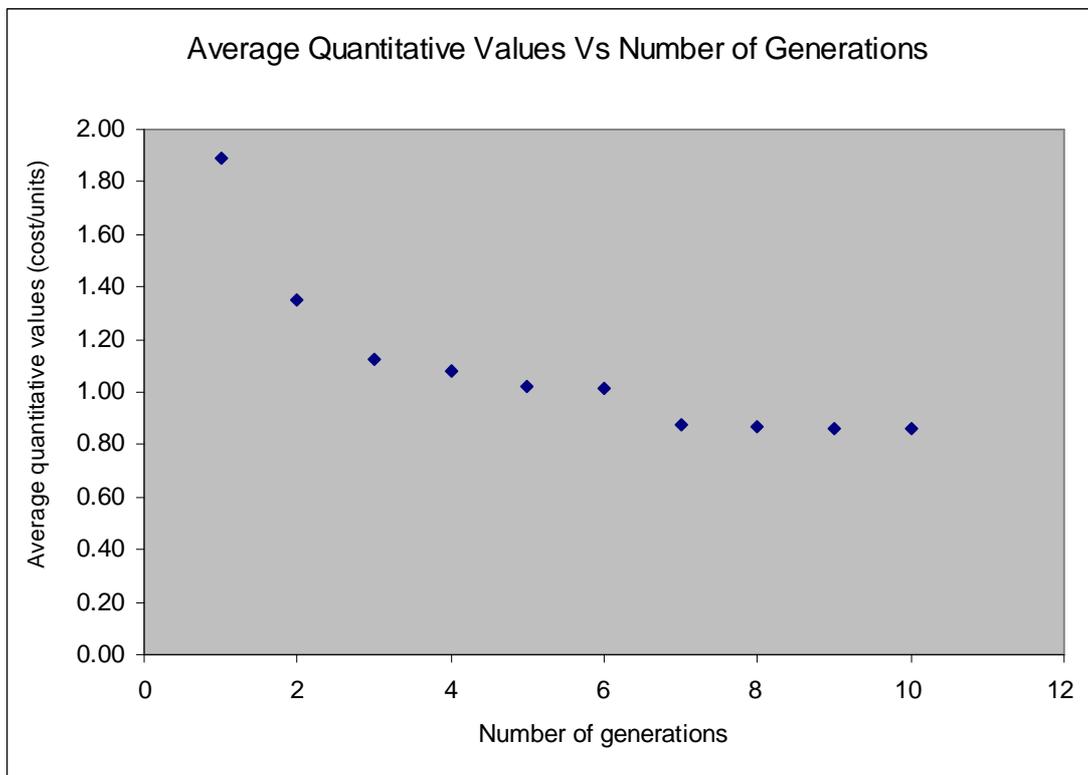


Figure 8.12: Quantitative fitness in manufacturing plant layout design using interactive NSGA-II

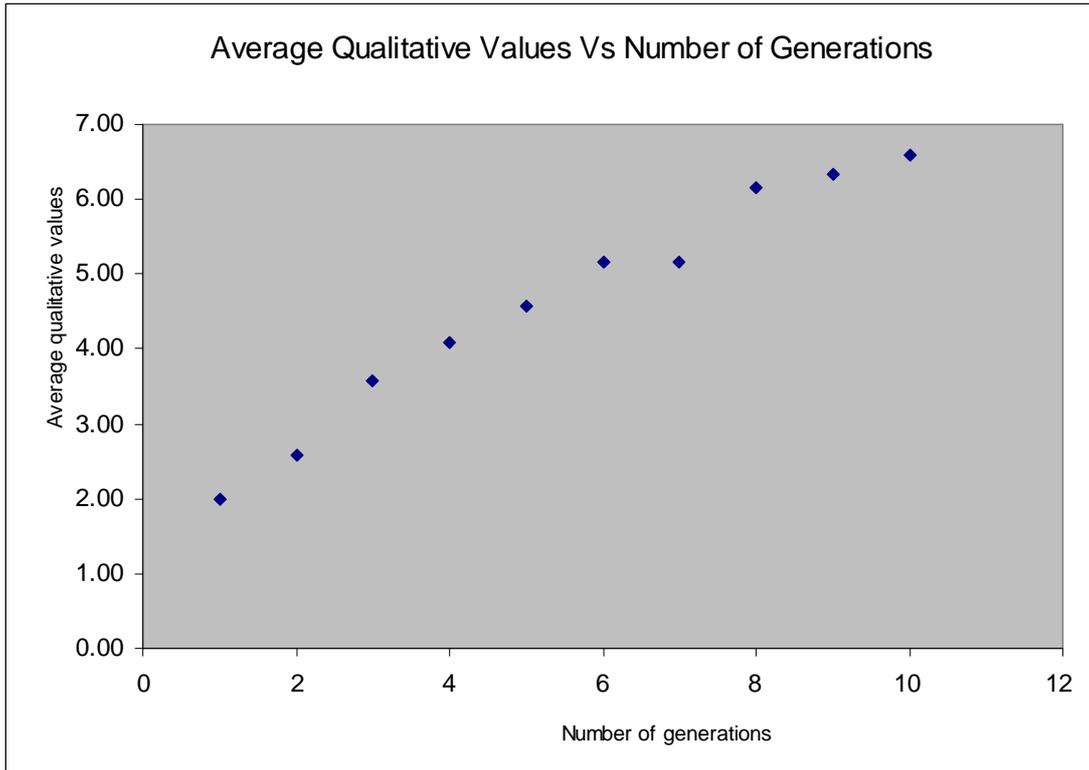


Figure 8.13: Qualitative fitness in manufacturing plant layout design using interactive NSGA-II

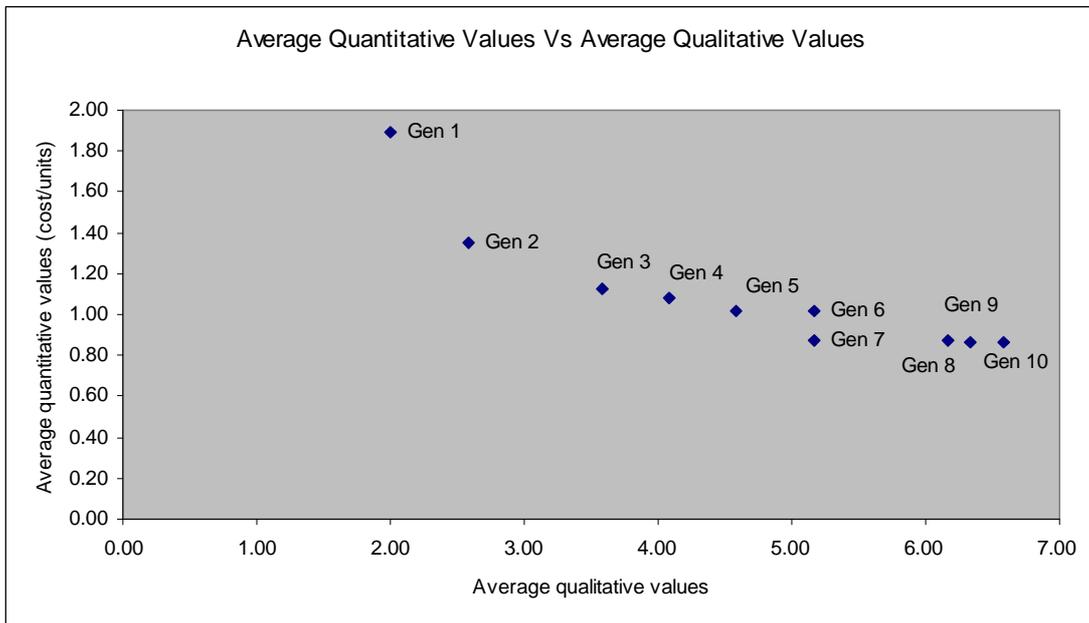


Figure 8.14: Average quantitative values against average qualitative values of manufacturing plant layout model (Gen=Generation)

Figure 8.14 shows a good convergence as the users gain more satisfaction from the designs produced with successive generations and the cost of building the plant

decreases. Figure 8.14 is the average quantitative fitness against average qualitative fitness of manufacturing plant layout model.

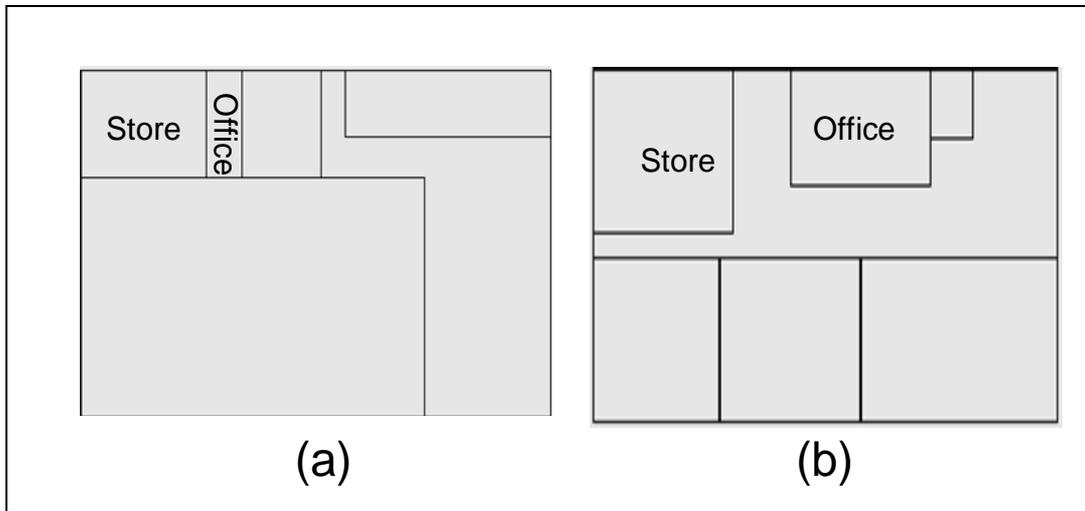


Figure 8.15: Two different designs obtained for the manufacturing plant layout model (Figure 8.15 (a) shows Store and Office too small and had poor rating of 3; Figure 8.15 (b) shows Store and Office having reasonable sizes and had good rating of 6).

In Figure 8.15, some sample designs that were generated are shown. Figure 8.15 (a) shows that the store and office are too small and so there is a low rating (3) for the design. Figure 8.15 (b) shows a bigger store and office, it has better rating (6) than (a). The grid provides the collaborative simulation of the quantitative and qualitative fitness functions. The visualisation of the same designs at the same time on different nodes by users help in determining the consistency and accuracy of the experiment. The floor planning model results are also obtained in the same manner as shown in Figures 8.16, 8.17 and 8.18.

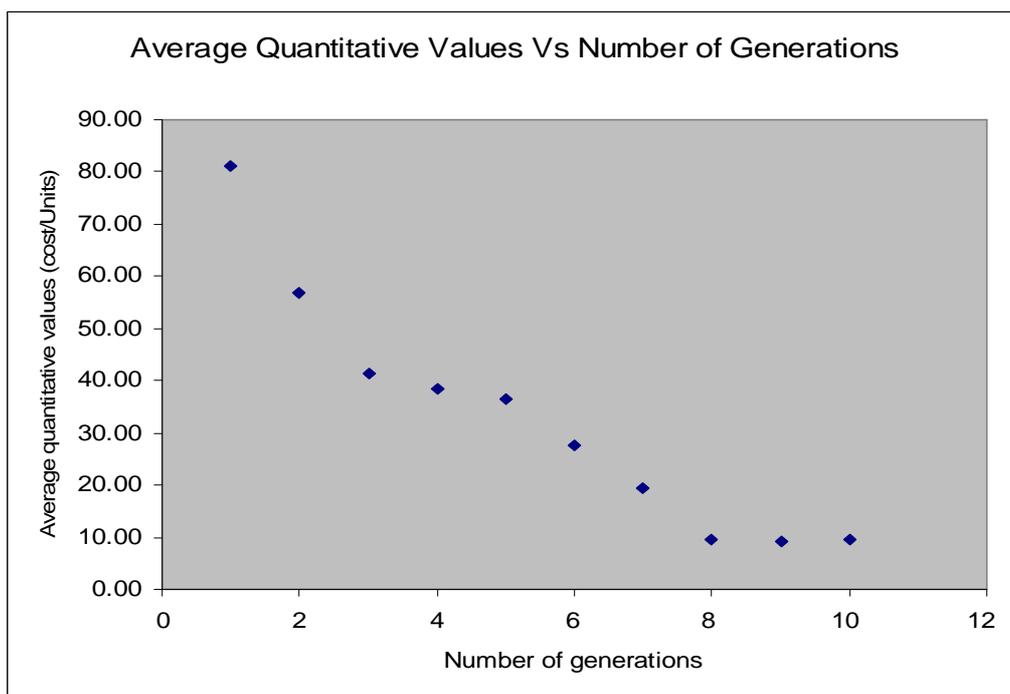


Figure 8.16: Average quantitative values (cost/units) with number of generations for floor planning model (Gen = Generation)

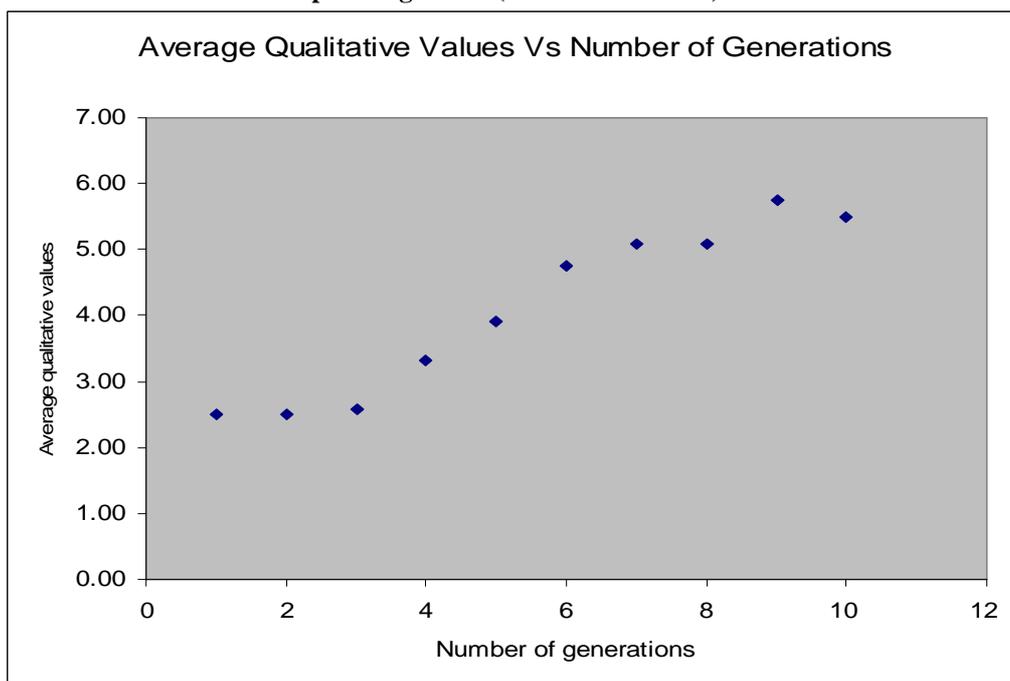


Figure 8.17: Average qualitative values (ratings) with number of generations for floor planning model (Gen = Generation)

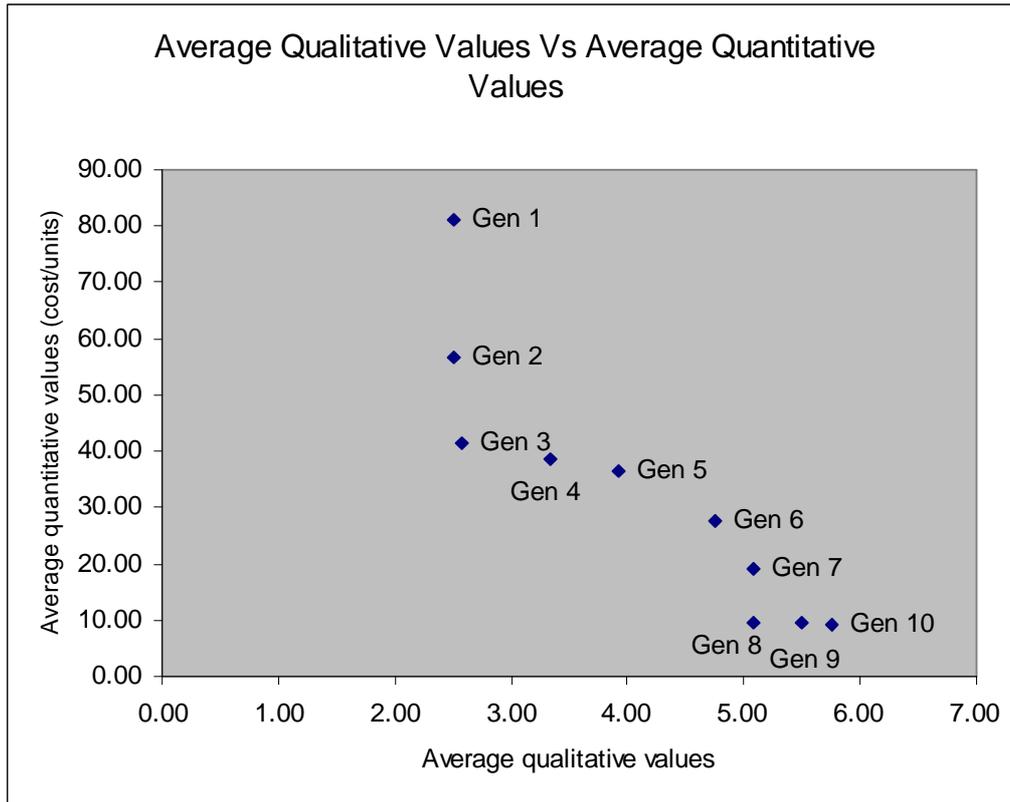


Figure 8.18: Average qualitative values and average quantitative values for floor planning model (Gen = Generation)

The discussion here is on the floor planning model. In figure 8.16, the cost of building the apartment decreases with successive generations. As the cost decreases, the designs generated improve and so the experts' ratings of the design increase. This explains why in Figure 8.17 the qualitative objective values (floor planning model) increases with successive generations. Figure 8.18 shows that as the quality of designs increases, the cost decreases. Although at a value between 2 and 3 of the qualitative rating, the cost decreases without much increase in the ratings (quality of design).

The multiple model optimisation process using two models using grid environment demonstrates how grid systems can be efficiently utilised to accommodate multiple processes. The migration of fitness functions and synchronisation of quantitative and qualitative evaluations were enhanced by the GridFTP while VOMS (Virtual Organisation and Membership Service) helped to assign specific roles and access to users based on who should access what at a time. This ensures trust and secure collaboration among distributed designers.

8.7 Validation of results

This section concentrates on discussing the applicability of the services implemented in chapter 7 with respect to the case studies. This is termed as verification. That is verifying if the system is working properly. The validation of the results using experts and confirmation of results from literature are also described. The discussion on the validation questionnaire is also part of the validation result.

This section will introduce the concept of single and multiple population parallelisations. When a model is subdivided (parallelised) into fitness functions so that they can be calculated at different nodes, the fitness functions need to produce values for each member of the population. There are two ways of doing this. The first method is to send each fitness function with a population. This amounts to duplication of the population and this incurs computational overhead. This is called multiple-population parallelisation method. The second method is to provide single population for the fitness functions to use. This is done by exposing the parameters and population to the fitness functions as a public module in the programming codes. This ensures faster and efficient computation. This is called single-population parallelisation. In this research, the single-population parallelisation of the NSGA-II optimisation process was adopted rather than the multiple population parallelisations. This is because single population parallelisation is easy to implement and use. This implementation was described in chapter 7. Multiple population parallelisations involve multiple communication and frequent migration of objective functions together with the parameters based on multiple populations.

The latency effect of running the optimisation of the three case studies in terms of CPU time is demonstrated. Figure 8.19 shows the effect of running the three case studies on one node of the system on the one hand and parallelising the computations on the grid platform on the other hand.

The turbine blade is more computational expensive than the other two case studies. For example its CPU time when run on single node is 525 seconds and reduced to 152 seconds when parallelised to nodes on the grid. The interesting aspect of Figure 8.19 is the results of the welded beam problem. It took approximately 5 seconds to run the

problem on a single node and 2 seconds on the grid. It is expected that it should have half a second or almost zero. This shows that it is not worth using the grid for problems that are not computationally expensive. This is because the time it takes for migration of fitness functions may delay the computation.

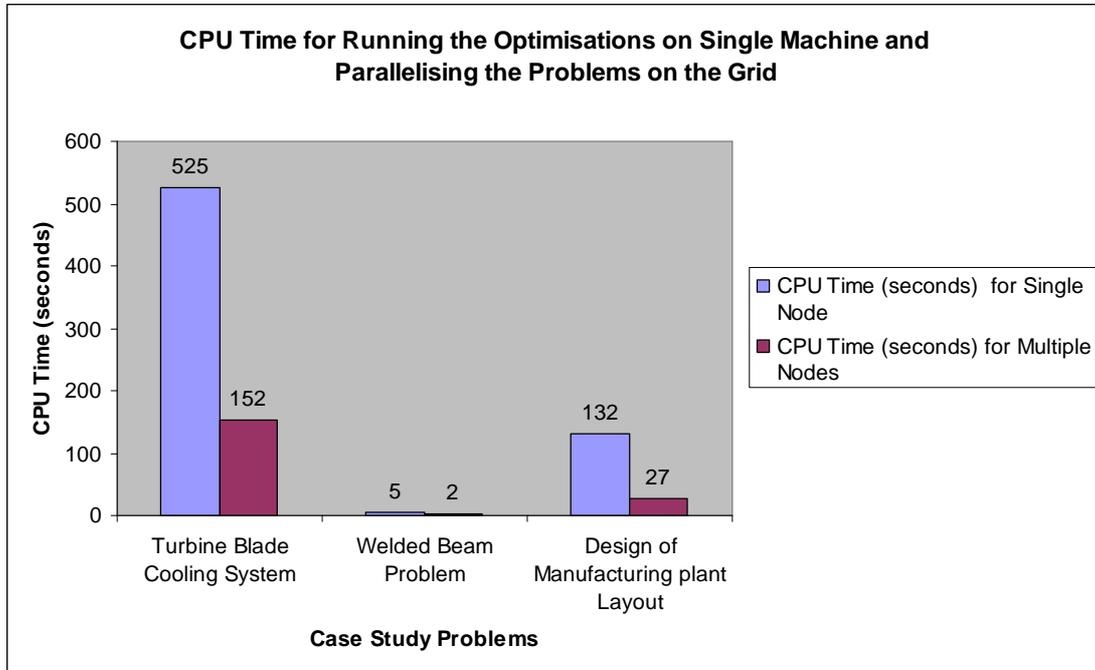


Figure 8.19: Time effect of running the problems on single node and on multiple nodes

Figure 8.19 is also discussed in section 8.7.1.1.

8.7.1 Why grid?

It is important to explain why the research chooses to use grid services for MODO. The following explanations are given in support of using grid computing for MODO in this research.

8.7.1.1 Computational synergy

From Figure 8.19, there is an improvement in the processing speed of the problems when the optimisation is run parallel on the grid platform as compared to running the problems on a single node. This shows the advantage and motivation of using the grid platform for MODO applications. The optimisation of the design of manufacturing plant layout used more CPU time than the welded beam problem. This may not be because of computational intensity of the problem, but the delay that might be caused by migration and interaction between the Q^T and Q^L objective functions. Again the

delay in passing the Q^L ratings to variables that make up the dimensions of the designs to produce subsequent designs which generate more data might have contributed to the time indicated.

8.7.1.2 Scheduling overhead efficiency in parallelisation and migration of processes

As explained earlier, the scheduling of resources among master and worker nodes as the optimisation process continues incurs overhead delays. This overhead delay is due to the migration of fitness functions from master node to different worker nodes for computation and storing the values of the fitness functions in a file and then transferring them back to the master file for operations such as crossover and mutation. This process continues up to the last generation. The computations are also performed for each individual in the population. Using the conventional FTP to run this process on the Linux environment, the CPU time for a population of 100 for one generation is approximately 1.5 seconds for the turbine blade cooling system case study. But when GridFTP was used for the same problem within the same environment but this time running Globus and the services developed, the CPU time was about 1 second for a population of 100. The difference may look negligible, but if one considers the cumulative effect of running thousands of populations for thousands of generations, it will be worth considering using GridFTP. GridFTP is more efficient in migrating huge files and data and scheduling resources in a distributed environment. The different CPU time obtained for the three case studies will be discussed later.

8.7.1.3 Running multiple models at a time

The grid as explained in literature is a decentralised resource sharing infrastructure. Multiple nodes can be used to run multiple models under different administrative domains. In the design of manufacturing plant layout case study, two models were run separately with each group having control to manage the process separately. This is possible by configuring Globus Resource Allocation Management (GRAM) to run on all the nodes. GRAM now manages the different resources. The capability of VOMS (Virtual Organisation Membership Service) discussed earlier was used to assign different user roles to the two groups. Access to the files and data were restricted and controlled by the each group. This demonstrates how the grid can be used by multiple users to run multiple optimisation applications. This helps in ensuring efficient

utilisation of computational resources of an organisation. The CPU time for the welded beam problem is negligible. The welded beam problem is used to demonstrate mathematical model building service and collaboration.

8.7.1.4 Decentralisation of control of optimisation and computational resources for reliability

In conventional HPC (High Performance Computing) systems, the resource management and scheduling systems are based on a single resource manager which is usually hosted by the server. In the event that the server crashes, all nodes are down making resources unavailable to all users. Another issue is that resources contributed by other users are not under control. In this research, the GRAM, which is the main resource manager and the Condor scheduler are installed and configured in the master node as well as in all the worker nodes. This means that all nodes have their individual resource managers and scheduling systems, even though the master still controls job submission attributes. In the event that the master node fails, other nodes can still function and communicate among themselves. The only problem is that the resources in the master node will become unavailable in the grid. The good thing is that when failed nodes are restored back to function, the resources are made available again as there is a capability to refresh services at certain interval of time.

In MODO applications, especially industrial applications, distributed designers may want to have administrative control over their resources and still collaborate with other MODO experts. This capability is accomplished in this research by configuring the Virtual Organisation Membership Service (VOMS) in Globus as explained earlier. VOMS enforces classification of users and privileges assigned to them based on their request when they join a grid community. This is where the user may agree for the resources he or she is contributing to be controlled centrally or only by him or her. The limiting aspect of VOMS is that it conforms to the single sign-on proxy authentication system in Globus certificate. VOMS has two main components namely VOMS which consists of the server for initialising the capabilities using the voms-proxy-init command and VOMS admin which is used to manage users and their privileges within the grid. These privileges are extended to the PostgreSQL database server to enforce user roles on the databases. This process is in conformity with the

idea of including SLA in the service specification document for providers and users of grid resources as roles are defined based on the service level agreement.

8.7.1.5 Flexibility in choice of search technique

The search strategy selection process described in section 7.5.3 enabled design experts to use the customized browser to query for search strategies that are available. The designer then views the property of the search strategy and decides which best suits his/her problem. This gives flexibility and instant way of choosing a search strategy. This capability is accomplished using the WebMDS and service group which display information on optimisation techniques and computational resources that are available. Users are further encouraged through the contents of the SLA (Service Level Agreement) contained in the service specification document (Figure 5.25) to subscribes for the search strategy based on the properties of the strategy as outlined in SLA.

The following sections of the validation describe the case studies. Each description considers the service used and optimization results.

8.7.2 Design of turbine blade cooling system

When the computation is done on only one machine, the Condor pull monitoring service indicated a low (15%) utilisation of systems. This is because some of the systems in the grid cluster were not being used most of the time. The researcher then flocked jobs to a pool of 3 computers when they were idle. The monitoring system then indicates a relatively high (30%) percentage of utilisation. This is a demonstration of how to make efficient use of idle systems in research centres and companies. From the users' point of view, using the input parameter GUI to enter design parameters and also make changes makes their life easier than using the Linux command prompt. The effect is that it increases efficiency as it is much faster and intuitive to conduct the optimisation.

In demonstrating the efficient use of computational resources, four different computations were used on four different nodes based on the three geometries and two or four objectives. One node performed the optimisation based on the plane geometry, another node based on the ribbed and the third node performs the

optimisation based on the pedestal geometry. A fourth node was used to perform the optimisation for two objectives problems for all the three geometries. Each user is able to view the results of all the 4 optimisation options (plane, ribbed, pedestal and two objectives) and compare the results. Again the Condor monitoring system showed an appreciable utilisation of computational resources. This method of carrying out optimisation and allowing distributed experts to view results is capable of increasing efficiency through collaboration and decision making.

8.7.3 Design of the welded beam problem

This case study is used to demonstrate the mathematical model building service. This is because the model is an explicit model. To do this, the task of building the model was divided among 5 users sitting at different nodes of DECGrid. One user built the cost objective function, another built the deflection objective function and another created the first two constraints. The last two constraints were built by the fourth user. The fifth user merged these components of the model and linked it to the problem definition file of NSGA-II then enters the parameters using the parameter input interface. The parameters include population size, number of generations, number of constraints and so on as described in Figures 7.12, 7.13 and 7.14 in chapter 7. The user then runs the NSGA-II optimisation service and the result obtained in Figure 8.6 appeared.

The results obtained compared favourably with those in literature (Tiwari, 2001). The model that was built also corresponded with the model in section 8.5. The participants at the validation process also confirmed that the results and mathematical model obtained are similar to the results from literature. The process of using the steps defined in the service specification to build the model and run the optimisation accomplishes the first, second and fifth objectives. This is because the grid requirements identified in literature and industry were designed and implanted to provide functional services for MODO application in the area of mathematical model building and optimisation together with the validation of the results in literature.

8.7.4 Design of a manufacturing plant layout and floor planning

Though this case study has human factors involved in the rating of the designs, the results obtained in Figures 8.12, 8.13 and 8.14 for the manufacturing plant model and Figures 8.16, 8.17 and 8.18 for the floor planning model are similar to the results obtained by Brintrup (2007) and Sushil (1993). This case study demonstrates how to include expert opinion interactively. The cost objective has a more regular graph (Figures 8.12 and 8.16) than the qualitative objective (Figures 8.13 and 8.17). This is understandable as the cost function is computed quantitatively while the qualitative satisfaction rating is given by experts. The expert may not be all the time consistent and that accounted for the nature of the graph. The collaboration service is used here by the experts rating the designs. As each expert visualises the designs, the objective evaluations are submitted to the database and file to form a stack. Each value in this stack is passed to the qualitative objective in the problem definition file of NSGA-II one after another corresponding to the values of the cost function. The ranking, crossover and mutation take place to produce the next designs. For successive designs, the computational intensity increases as the time to visualise the designs becomes longer. This time gets shortened if the optimisation process is parallelised to idle nodes on the grid. The demonstration of the computational service to speed up the generation of the designs for 5 runs is shown in Table 8.8.

Table 8.9: Average time comparisons for designs generated between one and three nodes

Run	No. of new designs	Total no. of designs	Average time taken to produce designs on one node for the models (Seconds)	Average time taken to produce designs when process is uses processors of 3 nodes (Seconds)
1	12	12	4.12	2.01
2	12	24	5.34	2.32
3	12	36	6.12	2.67
4	12	48	6.85	2.92
5	12	60	7.21	3.13

Table 8.9 shows that the number of new designs added to successive runs is 12. This corresponds to the number of population as described earlier. For the first run, the average time taken to produce the designs is 4.12 seconds using a single node and 2.01 when the processors of 3 nodes are used. This is a huge improvement if very high numbers of runs are considered for more complex optimisation problems.

8.7.5 Validation questionnaire discussions

Experts that participated in the validation filled a questionnaire (see Appendix-I). The questionnaire has 7 questions and summary of respondents' answers to them is discussed in this section. The first question asked the validation participants if the system achieved the aim and objectives of the research. The researchers all agreed that it achieved the aim and objectives. One of the users said that the interface used could be improved to make it friendlier. The second question asked for the limitations of the system. The respondents shared the opinion that the system is not fully automated as it involved user involvement in some of the steps and that the system is not tested in an industry and used only NSGA-II to test the case studies. A more automated system is suggested to improve the system as in answering the third question. The system is generally agreed by the researchers to be useful as it produced good results compared to the results in literature. The respondents also agreed that the system allow users to perform MODO in distributed environment and also provide essential components for designers to perform MODO. During the demonstration of the mathematical model building service, the models built corresponded with the models in literature. It is agreed that the system contributed to knowledge by providing service specifications and framework that leads to implementation of MODO application.

8.8 Summary

This chapter described three real-life case studies and their implementation. The case studies are the design of turbine blade cooling system, design of the welded beam problem and design of the manufacturing plant layout/floor planning. The results obtained were validated from results in literature. This is done using the validation methodology proposed in section 8.3. The validation is done with regards to the services implemented in chapter 7. The next chapter will discuss the whole thesis and give comments as conclusions.

Chapter 9 - Discussion and Conclusions

This concluding chapter presents the discussions on the key areas of the research and proposes future research issues. The discussion also highlights the generality of the research as well as its limitations. The chapter finally closes with concluding remarks and recommendations. The main focus of this chapter is:

- *To briefly discuss the key observations from this research*
- *To state the main contributions of this research*
- *To summarise the limitations of the research and*
- *To propose future research issues from the research*

9.1 Discussion

This section concentrates on discussing key observations of the research. Grid computing has been in the research domain for over two decades now and users want to see its practical applications in solving real-life problems. The delay in realising this goal is almost turning the grid concept into hype. This has also given way to some other notions known as cloud computing which is the utility concept of the grid coined from electricity grid. It is with this realisation that this research embarked upon developing grid services for solving MODO problems. MODO problems have traditionally been solved using high performance computing. This has helped in some ways to overcome the computational needs of the process. Additionally, user friendly interfaces, collaboration services and the intuitive guides to carry out optimisation bring efficiency into the MODO process. This research has designed and implemented mathematical model building service, collaboration service and optimisation service. In addition, computational service is provided for MODO experts to submit jobs for computation. The mathematical model building service enables designers to follow a step-by-step process to build a model. The first step is to select the problem domain and the system guides the designer through the creation of criteria, design parameters and constraints. This process can be carried out by distributed experts in different fields. The collaboration provides an interface for designers to share resources such as data and information from distributed environments. The collaborative process of building mathematical model has the advantage of triggering innovation and creativity

among collaborating designers. The optimisation service provides the designers with NSGA-II as a service for and an interface to enter parameters to run the optimisation. The computational service provided allows users to submit jobs to distributed nodes to speed up the optimisation process. The advantage of providing resources as services is efficient resource sharing. This means that users do not need to own all resources they need for optimisation as they can share their collective resources. The key observations of this research and the achievements of objectives of the research in each chapter are discussed below.

9.1.1 Literature review

The research conducted a literature review into grid computing applications in general and for MODO applications in specific. This review revealed the trend of evolution of grid research and focus. This trend is recorded by De Roure *et al.* (2003) as first, second and third generations of grid computing. This research has extended this to fourth and future generations of grid computing. Foster *et al.* (2001) also recorded this trend with regards to grid component technologies and standard protocols. This classification of the trend in grid evolution gave the research a focus on where grid services were used, are concurrently being used and what the future holds for grid research and applications. From the review, data and computations dominated the need for the first generation of grid applications. The I-WAY and Globus projects were some of the early test beds to demonstrate this concept of computational synergy obtained from connecting supercomputing centres together in San Diego (Foster and Kesselman, 1999). The second generation concentrated on solving the interoperability problem arising from the linking of various centres with each having different heterogeneous hardware and software applications. The middleware concept came up as an important research focus to allow heterogeneous systems to communicate among the grid community. It is worth mentioning that each successive generation was motivated by challenges faced during implementations. The third generation focused on problem solving environments for application areas and fourth generation looked at the service-oriented request and provision of grid functionalities as services. This concept is appealing to researchers as capabilities of grid can be shared by uses as services using the OGSA and WSRF platforms and interfaces. Future grid research issues look at ubiquitous grid computing, autonomic grid computing and cloud computing concepts. This research uses the PSEs and grid services concepts to design

and develop a framework for MODO applications. The review then looks at different MODO methods such as classical and evolutionary algorithms. The disadvantages of classical methods in using the parallel architecture of grid and the inability to produce multiple optimal solutions in a single simulation run were discussed. These disadvantages highlighted the reason why the research decided to use NSGA-II for the optimisation. Related MODO PSEs such as GEODISE, FIPER, SORCER and DAME were studied. The design of these PSEs and the services they provide were studied and gaps identified in them. The tools for deploying grid applications were identified in the review. This helped in avoiding problems during implementation.

The literature review helped to achieve the first objective by studying related problem solving environments and understanding the requirements for grid environment to solve MODO problems. From the literature gaps were identified and this formed the basis of the aim and objectives of the research. The strengths of the literature are the rigorous identification of the trend of evolution of grid computing with their classification and the discussions on MODO methods and approaches. This helped in giving a focus to the research. The likely sources of bias in the literature are the concentration of the study of problem solving environments that are European and American sponsored. However, this is not deliberate because majority of grid projects are implemented in UK, few European countries and the USA. Again most of the projects are sponsored by governments.

9.1.2 Industrial context

The idea of this research is to develop a framework to be used by practitioners in the MODO field. To do this, an industry survey was carried out to give the research an industrial context. The researcher visited the department that is in charge of MODO research using web and grid services in BAE Systems, a UK aerospace company and interviewed the head of the department and middleware programmer. It is gathered that the company does not use the popular Globus middleware system. The reason is that the proxy security feature allows third parties to give access to users. Being a defence company, they decided to develop an in-house middleware called GRIA (Grid Resource for Industrial Applications) which restricts authentication and authorisation of users to the owner of the resource based on bipartite security feature and not single-sign-on proxy. The Schlumberger Information Solutions (SIS)

department in Schlumberger Oil Company UK was also visited by the researcher. Here the company appreciated that any increase in computational power will have significant impact on making good decisions on oil well exploration prospects and on increasing the profitability. The researcher used the opportunity to discuss the concept of cycle-stealing using Condor. The company has developed an in-house seismic data simulation package called Eclipse. This software is used by many oil and gas exploration companies across the world. The research learnt that intuitive interfaces are good features for optimisation engineers and Eclipse tries to enhance this feature in its new versions. Eclipse runs on Linux cluster. A telephone interview was conducted with the Microsoft Technology Officer in UK on what grid technology issues the company is addressing. The company has released its first Compute Cluster Server. This is meant to work like Linux, but with GUI to send jobs for parallel processing. The latest version is Windows 2008 Compute Cluster Server with some enhanced features for grid middleware. An online questionnaire conducted by the researcher and received 21 responses from about six countries. This gives the research an international input as respondents from UK, USA, France, Canada, Denmark and Nigeria filled the questionnaire. The respondents are from research centres, universities and companies. Majority of the researchers were doing research in computational fields such as particle physics, bioinformatics and multi-objective optimisation. The respondents from companies were from aerospace and banking sectors. The responses were interesting as information was gathered on which areas applications are useful for grid and why. Most of the respondents said that they are using grid for data and computation and majority are using Globus middleware with Matlab for visualisation. Majority of users said that they deployed their grid systems on Linux operating system with reasons such as free download, open source and flexibility to amend the codes. The researcher also participated in a 6-week internship with Intellect, a division in the Department of Trade and Industry (DTI). The researcher spent the first one week at the National e-Science Centre (NeSC) Edinburgh. At NeSC the researcher had discussions with the training group, metadata group and middleware group. Issues such as metadata workflow dependencies and support for early grid systems were discussed. The researcher interviewed 2 people during the internship.

The industry survey helped to crystallise and achieve the first and second objectives of the research. This is achieved by comparing findings from literature and industry to come up with requirements that both suit the research and industry community. The idea of including graphical interface for entering optimisation parameters and building mathematical model is to satisfy usability requirements for designers. The strengths of the industry survey is the use of 3 three methods (interview, observation and questionnaire) to gather data. Additionally, the online questionnaire added an international appeal to the findings as respondents came from other countries such as the USA, UK, France, Canada, Denmark, Germany and Nigeria. The likely source of bias in the industry survey is the respondents that are from commercial companies that have business interest in grid technology and may promote the concept more than expected. The weakness in the industry survey is that only 3 companies were interviewed and they are all based in the UK. This is part of the reasons why the researcher conducted an online questionnaire to reach out to other companies and countries.

9.1.3 Gaps in literature and industry

The gaps in literature and industry survey are summarised in Table 4.2 in chapter 4. This table highlighted key application areas and concerns in literature and industry. For example, the key concern in both literature and industry is security of grid systems. As said before this is not part of this research but is worth discussion. The concern of industry is the single sign-on proxy security feature of Globus middleware. This feature allows third parties to give access to users. Companies want to have sole permission to give any user access to their data. The GRIA (Grid Resource for Industrial Application) middleware concept is to address this issue. GRIA uses what is called bipartite authorisation feature which restricts the right to give access to grid resources to resource owners only. The researcher is of the opinion that middleware should address both research and industry requirements. The research community uses open source standard systems such as Linux operating system and MySQL database while industry uses both open and proprietary systems. This is to done usually to satisfy their business rules. Industry needs resource brokers that incorporate accounting and metering features for application of grid systems. The research community is more concerned with increase in computational power and data storage during implementation while industry is also worried about these two as well as

integration of its business processes. This is because companies want to identify the economic value of any systems they use.

Though a lot of research is going on in MODO areas using grid computing, literature reveals that there is no attempt to provide service specifications document that end users can use to subscribe for grid services. The researcher enquired from the developers if they have ever seen a service specification document, apart from the XML-based specifications for software deployment. Nobody could provide this and even during industry survey the researcher could not get one. This made the researcher to analyse the designs of selected PSEs and come up with a service specification document. This service specification document consisted of steps for service level agreement between service providers and service requestors. This is to ensure that the end user of any grid implementation is considered and quality of service is ensured. The process of grid service definition which is lacking in literature and industry survey was also identified and proposed. Though FIPER has the capability to produce mathematical models, this research devised an intuitive method of creating the criteria (outputs), constraints and linking the criteria to the design parameters (inputs) to build a mathematical model. This model is linked to the NSGA-II optimisation service.

9.1.4 Design of DECGrid

The design of DECGrid is obtained from the gaps identified in literature and industry survey. The design helps in identifying the process of MODO and then aligning it to the capabilities of grid. To come up with the designs, the researcher studied the design of related problem solving environments. These included Geodise, FIPER, SORCER and DAME. A comparison and analysis of these designs was made and the researcher used the knowledge to design MODO services for DECGrid. The design looks at the mathematical model building service, collaboration service, computational service and design parameter input/optimisation service. The Unified Modelling Language (UML) is used for the design to show the use case diagrams and sequence diagrams. An objected-oriented approach is used to describe different classes with their objects and methods. This is what was used for developing the framework. Each design of the services mentioned will be discussed briefly. The design of the mathematical model building service started with the use case diagram. This helped the researcher to

identify the actors and operations needed. The main actors identified are expert, user, Globus toolkit and design interface. The user defines the domain, criteria, parameters and constraints to build the model. The expert defines good and bad designs and then validates the model. A class diagram (see Figure 5.5) is provided to cater for the stages of building the model. The sequence diagram provided the detailed steps that accomplished the model building process. The process described is for the quantitative model. The research used qualitative model in the third case study. The main actors in the qualitative service are optimisation engineer, Qt Designer, NSGA-II and Globus toolkit. Qt Designer provides the visualisation of the designs and the input interface for the optimisation engineer to enter the rating (qualitative evaluation) for each design. NSGA-II used these ratings with quantitative values to perform operations such as crossover and mutation. The sequence diagram is also described. The main actors for the collaboration are PostgreSQL database server, resource owner, optimisation engineer Globus toolkit and Condor. The design for the compute service concentrated on using Condor to schedule computations. This is done by identifying idle machines and submitting jobs for them to process. The optimisation service provided parameter input interface and NSGA-II as a service. The summary of the complete design is represented in the service specification shown in Figure 5.25.

The service specification (Figure 5.25) begins with the negotiation between service provider and service requestor. This covers issues concerning availability of service, functionalities of the service, scalability of the service and interoperability of the service. Others are support and maintenance, service guarantee, service renewal and service termination. This stage ensures quality of service provided and is the baseline for service agreement between providers and users. This is important so as to protect users' interests. This feature also gives utility definition to the MODO services. After the agreement stage MODO resources are registered in the resource registry and get aggregated in the aggregator source. A search strategy selection interface is provided for MODO users to query for available search strategies. The four services described earlier are then shown as part of the service specification documentation. This document is basically divided into three parts. These are quality of service assurance agreement, technology that host MODO resources and the specifications for the MODO services provided. This is the process used to obtain the framework and architecture described in chapter 6.

The design which is found in chapter 5 helped in achieving the third and fourth objectives. This is because the service specification document (see Figure 5.25) provided the specifications that are used to come up with a framework and architecture that supported the development of the MODO services. The framework (Figure 6.1) describes the step-by-step process that a grid service provider needs to follow to implement a grid service. The steps are identification of service requestors, choosing a grid service, define service requirements, identification of other services and developing the grid platform. The next stage is for the provider to agree on terms of services with the user after which implementation commences. This step ensures that the right users are identified and the right services are provided for them. This is the principle was used in the implementation of DECGrid. The strength of this chapter is the detailed designs provided for each service. The UML designs are easy to understand by both technical and end users of the services. The inclusion of service level agreement in the specification made the design a truly utility infrastructure that considered the end users as important part of the design. This helps in describing issues of quality of service provided. The process of providing grid services also helped to avoid problems of implementation as competing alternatives of providing grid services and the requestors are part of the process. The bias and limitation in the design is that it is tailored towards satisfying the requirements of MODO users. Service specifications in other applications were not considered. This is likely to affect the generality of the design. It is also not possible to obtain a design that can capture all scenarios.

9.1.5 Implementation of DECGrid

The design served as the road map for the implementation. The implementation followed the steps outlined in section 6.2 which is part of Figure 6.1. The steps in this process will be described briefly. The requestors identified are MODO experts. This is why the services implemented are specifically for MODO applications. The definition of grid service requirements were captured in literature and industry survey. These requirements were translated into designs in chapter 5. The researcher identified related problem solving environments such as Geodise, FIPER, SORCE and DAME. This helped in understanding the requirements for MODO applications. The

conventional platforms were identified in literature and industry survey. The main standard protocols are provided by the Open Grid Forum (OGF) documentations.

The architecture shown in Figure 6.2 is used for implementation. This architecture is the physically arrangement of the computers and the software needed to deploy the services. There is one server and seven clients. The server and clients have similar software such as Globus, Condor and Apache web server installed and configured on them. The Globus Resource Allocation Management (GRAM) is configured in all nodes to ensure autonomy of resource management. This makes resources on each node to run even if the server fails. The OGSA and Web Services Resource Framework (WSRF) enable services to run as web or grid services. The service provider registers resources on the server and the optimisation engineers subscribed the resources from distributed nodes. Even though the system was setup in one room, the researcher assumed scenarios for geographically distributed users. The quantitative model service, qualitative model service, collaboration, optimisation service and compute service are represented as published services.

The information obtained during literature review and industry survey was used to get the necessary tools for the implementation. The researcher initially wanted to use Windows platform to deploy Globus, but encountered problems. The problems were related to certification installation and GridFTP configurations. The researcher contacted members of the grid Globus toolkit forum and majority of the members used Linux. In addition to the problems, not all functionalities of Globus run on Windows. The researcher then opted for Linux. This shows that it is important to collaborate with grid users so that implementation problems can be avoided. Subscribing to research forum that used tools helped a lot. The researcher subscribed to PostgreSQL, Globus toolkit and Condor users' forums. This helped the researcher in overcoming installations and configuration problems. It also helped to be aware of new research topics in the grid community. There were also some problems with security configurations and host authentication for the collaboration service to allow distributed MODO users to share optimisation resources. This was solved using the PostgreSQL database configuration settings on each node. The heterogeneity in using different languages such as C (for NSGA-II), C++ (for Qt Designer that implements the qualitative design ratings of the third case study), PHP for the web interfaces and

java for the services in Globus proved to be difficult but was finally solved using the OGSA (Open Grid Services Architecture) interfaces. A room with 8 computers was dedicated for the implementation. The services mentioned in section 9.1.4 were implemented. The implementation as said earlier followed the steps described in the framework.

The implementation accomplished the aim of the research and put into practical use the first four objectives. This is because in implementing the MODO services, the researcher used the requirements captured for grid environment in literature and industry survey to come up with the service specifications (designs) and framework. The framework is implemented using the architecture. The novelty of the implementation is the process of building mathematical models and then linking them to the problem definition file of NSGA-II and running the optimisation as a service using the structured approach described in the framework. This process enables designers without optimisation knowledge to carry out optimisation, thus increasing the efficiency of design engineers.

The strength of the implementation is the workability of the services and the user interfaces. This means that the services were tested and used by researchers that participated in the validation. The weakness of the implementation is that it is implemented within Linux environment only. It will be good to have the Windows version of the implementation in future research as many design experts may prefer to use Windows. The choice of Linux environment became inevitable as the grid community which the researcher was interacting with could not help with Windows related problems. Thus it can be said that the grid community encouraged biased against commercial operating systems. There were some challenges during the implementation. It took many months to get the system working. This is because the researcher tried first to implement in Windows environment but faced difficulties in trouble shooting as many researchers that were contacted could help. The grid community is more comfortable with Linux. Besides, not all functionalities in Globus run in Windows. Two nodes of the system failed and had to be reconfigured which took a lot of time of the researcher.

The system is robust enough to be maintained. The services can be upgraded to meet up with new functionalities in new versions of middleware. This is because the system is built on open source system based on open grid standard architecture and is well documented.

9.1.6 Validation using real-life case studies

Three real-life case studies were used to validate the workings of the framework. The first case study-design of a turbine blade cooling system was used to demonstrate the computational service and collaboration service. The computational service used Condor-G to flock jobs to Condor pool which increased the computational speed of optimisation. The validation scenario for turbine blade used five researchers. Four researchers assumed the roles of resource owners of four nodes. The four objective functions were parallelised and submitted to each node for computation by the fifth researcher. The parameter input and NSGA-II optimisation service was used by the researchers to enter design parameters and then link the model to the NSGA-II optimisation service. The results obtained were compared with the ones in literature (Roy, 1997; Tiwari, 2001). The results were similar. This accomplished the fifth objective. This is because the research results have been validated using the turbine blade cooling system as a case study. The welded beam problem was used to validate the mathematical model building service, collaboration service and NSGA-II optimisation service. The model was built on 4 different nodes (2 objectives on separate nodes and 4 constraints on 2 nodes) and the input parameters were entered using the parameter service on another node and the optimisation was run using the NSGA-II optimisation service on yet another node. This showed how distributed users can access functionalities for MODO as services to carry out MODO processes. The results obtained corresponded to the results in literature (Tiwari, 2001). This case study validation accomplished the first two objectives. This is because the requirements for mathematical model service were used on the grid platform and allowed optimisation to be run to produce good results. The design of the manufacturing plant layout/floor planning showed how one node can serve for quantitative computations and another node for qualitative objective evaluations. This case study demonstrated an interactive NSGA-II scenario in which the users wait the designs to be visualised on any nodes of their choice. The users can then rate the designs. These ratings were then linked and merged to the cost evaluations in NSGA-

II for the next set of designs to be produced in the next generation. The case study also demonstrated that multiple models can be optimised by sharing grid resources and still can maintain decentralised control. Computational, collaboration and NSGA-II optimisation services were used here to bring efficiency to the process. The results obtained were compared with results in literature (Brintrup, 2007; Sushil, 1993). The results tallied with those in literature. This case study accomplished fifth objective. This is because the results obtained were the same with the results in literature.

The validation chapter achieved the last objective of the research. This is because the real-life case studies were used to validate the system. Experts participated in different scenarios during the validation. The strength of the validation is the results obtained. The results showed better convergence than results from literature. The bias in the validation is that majority of the experts are optimisation experts and only NSGA-II is used.

9.1.7 What benefits will business get from the system

This section looks at how this system can be implemented for industrial use. Although the system is not taken to industry for implementation, it is important to look at its features that are benefits to the business community. The provision of computational and optimisation resources as services is appealing for industrial use. Companies that use huge computational power for optimisation spent a lot of money on maintenance of hardware and software systems. Usually companies bear the cost alone. If computational resources are provided as services, companies can depend on service providers and pay for services per usage. Additionally, companies can collaborate among themselves to their computational resources together. Already Google, Amazon, IBM and Microsoft are beginning to implement such innovative concept called cloud computing. The user friendly interface and the intuitive steps provided to perform optimisation can increase the efficiency of designers.

9.1.8 Pending issues

There are pending issues if the system must provide business values to industry. The pending issues are beyond the scope of this research but they are worth discussing. The first issue is data security. As observed in literature and industry (see Figure 4.9), the most important fear raised by companies in implementing grid systems is security

of data and information. Although the Grid Security Infrastructure takes care of research security issues through the single sign-on proxy system, the industry community is not comfortable with this as third parties can give access to data on behalf of a company. This is why one of the companies visited during industry survey told the researcher that the idea of developing Grid Resource for Industrial Application (GRIA) middleware is to address industrial security issues. The opinion of this research is that the Globus middleware which is the most popular middleware (see Figures 2.3 and 4.5) among researchers and industry should include features that accommodate industrial security needs. The second issue is licensing. If companies must share computational resources as services, then the question of how services can be licensed must be devised. This research is of the opinion that a metering system likening electricity metering system could be devised to cater for this. This means that instead of licensing, users are charged per usage based on parameters such as time, speed or amount of storage or bytes of data/service.

9.1.9 Main contributions

The main contributions of this research are summarised below.

- Service specification document (Chapter 5). This research has proposed a service specification document for MODO applications. This service specification document consists of binding agreements between providers and requestors. This is to ensure quality of service to end users. In addition, the service specification document contains customised grid functionalities that allow MODO resources to be registered and subscribed. High level descriptions of the class diagram of services provided are also part of the service specification document. This is a contribution because the service level agreement ensures that service requestors get good quality of services. This is novel because research and industry survey revealed that related service specifications only have the class diagrams for implementation by programmers without the service level agreement component.
- Framework for implementation of MODO services (Chapter 6). The framework proposed consists of the process for service definition. This means that there is a step by step process to define a service. The process provided a road map for service providers to implement services that satisfy the needs of users. The process enables service providers to avoid implementation problems. This is a

contribution because literature and industry survey did not reveal any structured approach to implementing MODO services.

- Grid architecture for MODO applications (Chapter 6). The architecture proposed for MODO applications is devised based on the service specifications and framework. The architecture makes some nodes specialist on quantitative evaluation and some on qualitative evaluation. This allows distributed experts to collaborate based on different specialities. This is a contribution because different experts have the opportunity to collaborate and yet have autonomy over their administrative domain. The novelty is the concept of making nodes as dedicated for qualitative and quantitative evaluations. This has the advantage of registering only the resources that are needed by design experts in the nodes and making them accessible to other designers.
- The process of building mathematical model and the schema for the model building service (Chapter 7). The step-by-step process of building mathematical model provided an intuitive and easy way to develop a model. The procedure starts by defining the domain and then guides the user to create the criteria, define design parameters, define constraints and link the criteria to the parameters. The schema designed in such a way that the design variables and parameters are saved to PostgreSQL database as well as text files, allowing modification and reuse of these data later on if needed. This is a contribution because the procedure for creating the mathematical model is not found in literature and the schema forms a template for future reference by users. The novelty is in the steps to produce the model.

The major difference between the system in this research and other problem solving environments such as Geodise, FIPER and SORCER is in the inclusion of service level agreement which ensures enforcement of quality of service and gives assurance to the end user in service specification document. The second difference is the structured process that a service provider needs to follow to implement MODO services. This is included in the framework. The last difference is the process of creating mathematical model using model building service. Because of these differences, this research is proposing a name for the system to join the league of problem solving environments. The name is Multi-Objective Design Optimisation Computing Environment (MODOCE). This is because the system provides the

functionalities for MODO applications using grid computing services in different ways from related systems.

This research demonstrates generality as the 3 case studies used are drawn from different application domains. Figure 7.2 in chapter 7 shows how more disciplines can be incorporated as new domains. The WebMDS browser and service group has the capability to publish classical and EC algorithms. The reason for using EC technique is to take advantage of parallel computations of the grid architecture. The industry survey conducted helped in generalising the research findings. This is because different companies with different backgrounds were visited. These companies were in the oil and gas, aerospace and software areas. In addition, the multiple methods of gathering data added to the generality of the research. The research used interviews, questionnaire and observation method. The online questionnaire reached out to respondents in about six countries. This further generalised the findings with international inputs. The implementation architecture is generic as server-client (or master-worker) implementation approached was adopted.

9.1.10 *Limitations of the research*

There are some limitations that are observed in the research. They are summarised below.

- The case studies used are only for engineering design optimisation. Other domains are not included.
- The research only used NSGA-II for the optimisation and the search strategy selection process was only implemented for stochastic algorithms.
- Due to time constraints and access restrictions the prototype could not be taken to industry for validation.
- The prototype is not used as a production grid; it is only tested and used for research purpose.
- The grid requirements that were used to achieve the first and second objectives were biased towards providing MODO functionalities.

9.1.11 *Future research issues*

Based on the limitations and other observations in this research, some future research issues have been recommended for consideration. These are discussed briefly below.

- The implementation of the main components of service specifications and framework as research topics on their own is a future research area. For example, service level agreement and the legal framework should be looked into. This is what will ensure quality of service to requestors.
- The process of service definition as stated in the framework can be redefined to cater for different application areas, outside MODO.
- The Globus proxy feature should be improved upon to cater for both industrial and research applications.
- The research can be improved by incorporating classical and other EC techniques to run concurrently on different nodes. Not all problems are suited for EC techniques.
- The case studies in the future should include disciplines other than engineering design optimisation.
- The validators in future research should include validators that are not knowledgeable in multi-objective design optimisation.

9.2 Conclusions

This section highlights the achievements of the research and reconciles them with the aim and objectives stated in sections 3.1 and 3.2 in chapter 3. The following discusses each objective.

- First objective: To understand the requirements from grid environment for multi-objective design optimisation (MODO). This objective was achieved by studying the service specifications and requirements of related problem solving environments (PSE) in literature and industry survey. Such PSE includes Geodise, FIPER, SORCER and DAME. The knowledge of MODO methods obtained from literature and industry survey also helped in accomplishing the objective. The literature produced a trend of evolution of grid computing. This evolution was classified as first, second, third, fourth and future generations. This classification described the trend of changes in grid requirements which also helped in shaping the requirements that was used for MODO. The challenges and special features of MODO applications such as mathematical complexity, collaboration and computational intensity guided the design for the requirements. The conclusions drawn in this research as regards this objective are (1) the objective helped in the

design of MODO services to suit grid requirements and (2) the method used to accomplish the objective is biased towards MODO applications.

- Second objective: To develop functional grid service specifications for MODO. This objective was accomplished by studying the special features of MODO in literature and industry survey. Features such as mathematical complexities, computational and data intensity and mixed variables and objective functions were studied. MODO is also collaborative and multidisciplinary in nature. The functional requirements developed in this research are mathematical model building service, collaboration service, optimisation and computational services. These services were designed using the Unified Modelling Language (UML) notations. The use case and sequence diagrams provided specifications that were useful for the service implementation. The friendly user interface ensures usability of the system by designers. The open grid services architecture and web services resource framework protocols built in Globus toolkit enabled the grid service specifications to be implemented. The conclusion for this objective is that MODO characteristics necessitated the development of the services implemented to cater for mathematical model building, collaboration, optimisation and computation.

- Third objective: To develop a grid architecture to support the MODO service. The third objective was achieved by identifying the process of defining a grid service and tools and facilities required to implement the architecture. The process of defining a service begins with the identification of service users. This is important as it helped in narrowing the requirements to MODO applications. The sequence in this process included service level agreement between providers and requestors. This ensures quality of services provided by the architecture. The factory pattern model is used to implement the services. This has the advantage of putting data separate from mathematical operations. The architecture provided a platform for providers to publish optimisation and computation resources as services. Distributed designers can then subscribe for these services and collaborate as well. This ensures efficient utilisation of resources. The conclusions are (1) the architecture was validated by researchers (2) the architecture has features that can accommodate implementation of services in other disciplines.

- Fourth objective: To develop a step-by-step process that a MODO service provider needs to follow to implement the architecture. This objective is accomplished by identifying the requirements and challenges of grid implementation in literature and industry. This process helped in avoiding problems of implementation. This process is obtained by interviewing MODO and grid experts in research and industry. The step by step process formed the first stage of the architecture proposed. The implementation of the architecture is based on Globus middleware and Condor scheduling system. The conclusions on this objective are (1) the steps helped the researcher to avoid implementation problems and (2) the steps have generalisation to accommodate implementation of other services.

- Fifth objective: To validate the research results using case studies. This objective is accomplished by using three case studies to validate the results produced. The three case studies are the turbine blade cooling system; welded beam problem and manufacturing plant layout/floor planning. A validation methodology was proposed and scenarios to validate the three case studies were identified and used. The researcher used 2 senior and 3 junior researchers for the validation. The results obtained were similar to the results in literature. The researchers who participated in the validation confirmed this in the questionnaire they filled. The conclusions are (1) 5 researchers participated in the validation (2) the validation was limited to research application and was not validated in industry.

References

Abbas, A. (2001), *Grid Computing: A Practical Guide to Technology and Applications* (2nd ed), Charles River Media, USA.

Abdelzaher, T. and Shin, K. G. (1998), End-host Architecture for QoS-Adaptive Communication, in: *Fourth IEEE Real-Time Technology and Applications Symposium (RTAS'98)*, p. 121-130.

Abramson, D., Buyya, R. and Giddy, J. (2002), A Computational Economy for Grid Computing and its Implementation in the Nimrod-G Resource Broker, *Future Generation Computer Systems*, vol. 18, p.1061-1074.

Al-Ali, R., Hafid, A., Rana, O. F., and Walker, D. W. (2004), QoS Adaptation in Service-Oriented Grids, *Concurrency and Computation Practice and Experience*, Vol. 30, p. 529-540.

Allcock, B., Bester, J., Bresnahan, J., Chervenak, A. L., Foster, I., Kesselman, C., Meder, S., Nefedova, V., Quesnel, D. and Tuecke, S. (2002), Data Management and Transfer in High-Performance Computational Grid Environments, *Parallel Computing*, Vol. 28, p. 749-771.

Aloisio, G., Cafaro, M., Fiore, S. and Mirto, M. (2004), The GRelC Library: A Basic Pillar in the Grid Relational Catalog Architecture, in: *Proceedings of the International Conference on Information Technology (ITCC'04)*, Vol. 1, 5-7 April 2004, Lecce Univ., Italy, p. 372- 376.

Aloisio, G. and Talia, D. (2002), Grid Computing: Towards a New Computing Infrastructure, *Future Generation Computer Systems*, Vol. 18, p. 5-11.

Amin, K., Laszewski, G. and Nijsure, S. (2002), Open Collaborative Grid Service Architecture (OCGSA), *Euroweb 2002: The Web and the GRID: From e-Science to e-Business*, Vol. 5, p. 1-7.

Andreozzi, S. Vistoli, C. and Sgaravatto, M. (2003), Sharing a Conceptual Model of Grid Resources and Services, in: *Computing in High Energy and Nuclear Physics*, Vol. 18, 24-28 March 2003, La Jolla, California, USA, p. 1-4.

Andrieux, A., Berry, D., Garibaldi, J., Jarvis, S., MacLaren, J., Ouelhadj, D., and Snelling, D. (2003), *Open Issues in Grid Scheduling*, Report Number UKeS-03, NeSC, Edinburgh, UK.

Andrzejak, A. and Xu, Z. (2002), Scalable, Efficient Range Queries for Grid Information Services, in: *Proceedings of the Second International Conference on Peer-to-Peer Computing*, Vol. 05-07 September 2002, Washington, DC, p. 33.

Antonioletti, M., Atkinson, M. B., Rob, B. A., Hong, N. P., Colins, B., Hardman, N., Hume, A. C., Knox, A., Jackson, M., Krause, A., Laws, S., Magowan, J., Paton, N. P., Pearson, D., Sugden, T., Watson, P. and Westhead, M. (2003), The Design and Implementation of Grid Database Services in OGSA-DAI, *Concurrency and Computation: Practice and Experience*, Vol. 17, No. 2-3, p. 357-376.

Antonioletti, M., Hong, N. C., Hume, A., Jackson, M., Krause, A., Nowell, J., Palansuriya, C., Sugden, T. and Westhead, M. (2005), Experiences of Designing and Implementing Grid Database Services in the OGSA-DAI Project, in: *Global Grid Forum Workshop on Designing and Building Grid Services*, p. 2.

Asiki, A., Doka, K., Konstantinou, I., Zissimos, A., Tsoumakos, D., Koziris, N., and Tsanakas, P. (2009), A Grid Middleware For Data Management Exploiting Peer-To-Peer Techniques, *Future Generation Computer Systems*, Vol. 25, p. 426-435.

Atkinson, M. (2003), Rationale for Choosing the Open Grid Services Architecture, in: Berman, F., Fox, G. C. and Hey, A. J. G.(editors), *Grid Computing: Making the Global Infrastructure a Reality*, Wiley Series in Communications Networking & Distributed Systems, p. 199-216.

Austin, J., Davis, R., Fletcher, M., Jackson, T., Jessop, M., Liang, B. and Pasley, A. (2005), DAME: Searching Large Data Sets within a Grid-Enabled Engineering Application, in: *Proceedings of the IEEE*, Vol. 93, 21 Feb. 2009, NY, USA , p. 496-509.

Baker, M, Apon, A., Ferner, C., and Brown, J. (2005), Emerging Grid Standards, *IEEE Computer Society*, p. 43-50.

Baker, M., Buyya, R. and Laforenza, D. (2002), Grids and Grid Technologies for Wide-Area Distributed Computing, *Software Practice and Experience*, Vol. 32 No. 488, p. 1437-1466.

Barbera, R., Falzone, A. and Rodolico, A. (2003), The GENIUS Grid Portal, *Computing in High Energy Physics*, Vol. , 24-28 March 2003, Carlifornia, USA, p. 1-8.

Barmouta, A. and Buyya, R. (2003), GridBank: A Grid Accounting Services Architecture (GASA) for Distributed Systems Sharing and Integration, in: *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS'03)*, Vol. 1, 1 Oct. 2003, p. 22-26.

Bell, W. H., Cameron, D. G., Carvajal-Schiaffino, R., Millar, A. P., Stockinger, K. and Zini, F. (2003), Evaluation of an Economy-Based File Replication Strategy for a Data Grid, in: *Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03)*, Vol. 10., 12-15 May 2003, DC, USA, p. 661.

Berlich, R., Kunze, M. and Schwarz, K. (2005), Grid Computing in Europe: From Research to Deployment, in: *Australian Workshop on Grid Computing and e-Research*, Vol. 44, Feb. 3 2005, Newcastle, Australia, p. 21-27.

Berman, F., Fox, G. C. and Hey, A. J. G. (2003), *Grid Computing: Making the Global Infrastructure a Reality*, John Wiley & Sons Ltd., USA.

- Bethel, E. W. and Shalf, J. (2003), Grid-Distributed Visualizations using Connectionless Protocols, *IEEE Computer Graphics and Applications*, Vol. 23, No. 2, p. 51-59.
- Bird, I., Robertson, L. and Shiers, J. (2005), Deploying the LHC Computing Grid - The LCG Service Challenges, in: *IEEE International Symposium on Mass Storage Systems and Technology*, Vol. , 21 sept. 2005, Swindon, UK, p. 160-165.
- Blythe, J., Deelman, E. Gil, Y. and Kesselman, C. (2003), Transparent Grid Computing: A Knowledge-Based Approach, in: *15th Innovative Applications of Artificial Intelligence Conference (AIAA-03)*, Vol. 12, 12-14 Aug. 2003, Acapulco, USA, AIAA.
- Boose, J. H. (1989), Using Repertory Grid-Centered Knowledge Acquisition Tools for Decision Support, in: *22nd Proceedings of IEEE Hawaii International Confercnce*, Vol. 23, 3-6 Jan 1989, Hawaii, USA, p. 211-220.
- Brintrup, A. M. (2007), *Handling Qualitativeness in Design Optimisation using Interactive and Multi-objective Genetic Algorithms* (PhD Thesis), Cranfield University, Cranfield, UK.
- Brockhoff, D. and Zitzler, E. (2006), Are All Objectives Necessary? On Dimensionality Reduction in Evolutionary Multiobjective Optimization, *LNCS*, Vol. 4193, p. 533-542.
- Brodlie, K. W., Duce, D. A., Gallop, J. R., Walton, J. P. R. B. and Wood, J. D. (2004a), Distributed and Collaborative Visualization, *The Erographics Association and Blackwell Publishing Ltd*, Vol. 23, No. 2, p. 223-251.
- Brodlie, K., Duce, D., Gallop, J., Sagar, M., Walton, J. and Wood, J. (2004b), Visualization in Grid Computing Environments, *IEEE Visualization*, Vol. 13, 10-15 Oct. 2004, Texas, USA, p. 155-162.
- Buyya, R. (2002), *Economic-Based Distributed Resource Management and Scheduling for Grid Computing*, (PhD Thesis), Monash University, Melbourne, Australia.
- Buyya, R., Abramson, D. and Giddy, J. (2001), A Case for Economy Grid Architecture for Service Oriented Grid Computing, *Proceedings of the 10th Heterogeneous Computing Workshop*, Vol. 2, Apri 5-6 2001, p. 83.
- Buyya, R., Jin, H. and Cortes, T. (2002), Cluster Computing, *Future Generation Computer Systems*, Vol.18, p. 5-8.
- Buyya, R., Yeo, C. S., Venugopal, S., JBroberg, J., and Brandic, I. (2008), Cloud Computing And Emerging IT Platforms: Vision, Hype, And Reality For Delivering Computing As The 5th Utility, *Future Generation Computer Systems*, Vol. 21, No. 6, p. 599-616.
- Cannataro, M. and Talia, D. (2003a), The Knowledge Grid: Designing, Building, and Implementing of an Architecture for Distributed Knowledge Discovery. *Communications of the ACM*, Vol. 46, No. 1, p. 89-93.

- Cannataro, M. and Talia, D. (2004), Semantics and Knowledge Grids: Building the Next-Generation Grid, *IEEE Intelligent Systems*, Vol. 19, No. 1, p. 56-63.
- Cannataro, M. and Talia, D. (2003b), Towards the Next-Generation Grid: A Pervasive Environment for Knowledge-Based Computing, in: *Proceedings of the International Conference on Information Technology: Computers and Communications*, Vol. 20, 28-30 April 2003, USA, IEEE, p. 437- 441.
- Cannataro, M., Talia, D. and Trunfio, P. (2001), Knowledge Grid: High Performance Knowledge Discovery Services on the Grid, in: *2nd Int. Workshop on Grid Computing*, Vol. 38, p. 25-31.
- Cano, J. R., Herrera, F., and Lozano, M. (2003), Using Evolutionary Algorithms as Instance Selection or Data Reduction in KDD: An Experimental Study, *IEEE Transactions on Evolutionary Computation*, Vol. 7, No. 6, p. 561-575.
- Cao, J., Jarvis, S. A., Saini, S. and Nudd, G. R. (2003), GridFlow: Workflow Management for Grid Computing, in: *Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID.03)*, Vol. 5, 12-15 May 2003, Augustin, Germany, p. 198- 205.
- Castillo, E., Conejo, A.J., Pedregal, P., Garcia, R., and Alguacil, N. (2002), *Building and Solving Mathematical Programming Models in Engineering and Science*, (4th ed), John Wiley, UK.
- Chadwick, D. (2005), Authorisation in Grid Computing, *Information Security Technical Report*, Vol. 10, No. 1, p. 33-40.
- Chao, K., Younas, M., and Griffiths, N. (2006), BPEL4WS-Based Coordination of Grid Services in Design, *Computers in Industry*, Vol. 57, No. 8, p. 778-786.
- Chen, E. J. and Lee, L. H. (2009), A Multi-Objective Selection Procedure Of Determining A Pareto Set, *Computers & Operations Research*, Vol. 36, No. 6, p. 1872-1879.
- Chen, L., Cox, S. J., Tao, F., Shadbolt, N. R., Puleston, C. and Goble, C. (2004), Empowering Resource Providers to Build the Semantic Grid, in: *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI'04)*, Vol. 9, 11 Aug 2004, DC, USA, p. 271 - 277.
- Chervenak, A., Foster, I., Kesselman, C., Salisbury, C., and Tuecke, S. (2000), The Data Grid: Towards an Architecture for the Distributed Management and Analysis of Large Scientific Datasets, *Journal of Network and Computer Applications*, Vol. 23, p. 187-200.
- Cheung, W. K., Liu, J., Tsang, K. H. and Wong, R. K. (2004), Towards Autonomous Service Composition in a Grid Environment, in: *Proceedings of the IEEE International Conference on Web Services (ICWS'04)*, Vol. 10, 6-9 July 2004, p. 550- 557.
- Chung, W. and Chang, R. (2009), A New Mechanism for Resource Monitoring in Grid Computing, *Future Generation Computer Systems*, Vol. 25, p. 1-7.

Cox, S. J., Fairman, M. J., Xue, G., Wason, J. L. and Keane, A. J. (2001), The GRID: Computational and Data Resource Sharing in Engineering Optimisation and Design Search, in: *Parallel Processing Workshop*, Vol. 10, 2-3 April 2001, Valencia, Spain, p. 207-212.

Cross, M., and Moscardini, A.O. (1995), *Learning the Art of Mathematical Modelling* (2nd ed), Ellis Horwood Ltd., UK.

Czajkowski, K., Fitzgerald, S., Foster, I., and Kesselman, C. (2001), Grid Information Services for Distributed Resource Sharing, in: *Proc. 10th IEEE International Symposium on High-Performance Distributed Computing (HPDC-10)*, Vol. 23, 7 Aug 2001, CA, p. 181-194.

Darema, F. (2005). Grid Computing and Beyond: The Context of Dynamic Data Driven Applications Systems, in: *Proceedings of the IEEE*, Vol. 93, No. 3, 3 March 2005, p. 692-697.

De Roure, D., Baker, M. A., Jennings, N. R. and Shadbolt, N. R. (2003), The Evolution of the Grid, in: Berman, F., Fox, G. C. and Hey, A. J. G.(editors), *Grid Computing: Making the Global Infrastructure a Reality*, Wiley Series in Communications Networking & Distributed Systems, p. 65-100.

De Roure, D., Jennings, N. R. and Shadbolt, N. R. (2005), The Semantic Grid: Past, Present, and Future, in: *Proceedings of The IEEE*, Vol. 93, No. 3, 2 March 2005, p. 669-681.

Deb, K. (2001), *Multi-Objective Optimization using Evolutionary Algorithms*, John Wiley & Sons Ltd., USA.

Defanti, T. A., Foster, I., Papka, M. E., Stevens, R., and Kuhfuss, T. (1996), Overview of the I-WAY: Wide Area Visual Supercomputing, *International Journal of Supercomputing Applications*, Vol. 10, No. 2, p.1-9.

Deelman, E., Kesselman, C., Mehta, G., Meshkat, L., Pearlman, L., Blackburn, K., Ehrens, P., Lazzarini, A., Williams, R. and Koranda, S. (2002), GriPhyN and LIGO, Building a Virtual Data Grid for Gravitational Wave Scientists, in: *Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing (HPDC'02)*, CA, p. 225-234.

Deelman, E., Singh, G., Atkinson, M. P., Chervenak, A., Hong, N. P. C., Kesselman, C., Patil, S., Pearlman, L. and Su, M. (2004), Grid-Based Metadata Services, in: *Proceedings of the 16th International Conference on Scientific and Statistical Database Management (SSDBM'04)*, CA, USA, p. 393- 402.

Dullmann, D., Hoschek, W., Jaen-Martinez, J., Segal, B., Samar, A., Stockinger, H. and Stockinger, K. (2001), Models for Replica Synchronisation and Consistency in a Data Grid, in: *Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing*, Vol. 0, p. 67.

- Dyer, C., Cordova, A., Mont, A. and Lin., J. (2008), Fast, Easy, And Cheap: Construction Of Statistical Machine Translation Models with MapReduce, in: *Proceedings of the Third Workshop on Statistical Machine Translation at ACL 2008*, Vol. 6, 19 June 2008, Columbus, Ohio p. 5-10.
- Eres, M. H., Pound, G. E., Jiao, Z., Wason, J. L., Xu, F., Keane, A. J. and Cox, S. J. (2005), Implementation and Utilisation of a Grid-Enabled Problem Solving Environments in Matlab, *Future Generation Computer Systems*, Vol. 21, p. 920-929.
- Eres, M. H., Pound, G. E., Keane, A. J. and Cox, S. J. (2004), User Deployment of Grid Toolkits to Engineers, in: *Proceedings of the UK e-Science All Hands Meeting 2004, 31 Aug - 3 Sep 2004, Nottingham, UK*, p. 424-429.
- Ferris, C. and Farrell, J. (2003), What are Web Services? *Communications of The ACM*, Vol. 46, No. 6 p. 45-57.
- Figueiredo, R. J., Dinda, P. A., and Fortes, J. A. B. (2003), A Case For Grid Computing On Virtual Machines, in: *Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS'03)*, Vol. 4, 5 May 2003, p. 456-466.
- Fleming, P. J. and Purshouse, R. C. (2002), Evolutionary Algorithms in Control Systems Engineering: A Survey, *Control Engineering Practice*, Vol. 10, p. 1223-1241.
- Flynn, M. (1972), Some Computer Organizations and Their Effectiveness, *IEEE Trans. Comput.*, Vol. C-21, p. 948.
- Folino, G. and Spezzano, G. (2007), An Autonomic Tool for Building Self-Organizing Grid-enabled Applications, *Future Generation Computing Systems*, Vol. 23, No. 5, p. 671-679.
- Forte, M., De-Souza, W. L. and Do-Prado, A. F. (2007), Using Ontologies and Web Services for Content Adaptation in Ubiquitous Computing, *Journal of Systems and Software*, Vol. 81, No. 3, p. 368-381.
- Foster, I. (2005), Service-Oriented Science, *American Association for the Advancement of Science*, Vol. 30, No. 8, p. 814-817.
- Foster, I., Geisler, J., Nickless, B., Smith, W. and Tuecke, S. (1998a), Software Infrastructure for the I-WAY Metacomputing Experiment, *Concurrency Practice Experience*, Vol. 10, John Wiley & Sons, Inc, p. 567-581.
- Foster, I., Insley, J., Laszewski, G., Kesselman, C. and Thiebaut, M. (1999), Distance Visualization: Data Exploration on the Grid, *Scientific Visualization*, Vol. 32, No. 12, p. 36-43.
- Foster, I. and Kesselman, C. (1999), *The Grid: Blueprint for a New Computing Infrastructure* (1st ed), Morgan Kaufmann Publishers, Inc, USA.
- Foster, I., Kesselman, C., Nick, J. M. and Tuecke, S. (2002a), Grid Services for Distributed System Integration, *IEEE Computer Society*, Vol. 35, No. 6, p. 37-46.

Foster, I., Kesselman, C., Nick, J. M. and Tuecke, S. (2003), The Physiology of the Grid, in: Berman, F., Fox, G. C., and Hey, A. J. G., *Grid Computing: Making the Global Infrastructure a Reality*, Wiley Series, p. 217-249.

Foster, I., Kesselman, C., Tsudik, G., and Tuecke, S. (1998b), A Security Architecture for Computational Grids, *ACM Communications*, Vol. 3032, p. 83-92.

Foster, I., Kesselman, C. and Tuecke, S. (2001), The Anatomy of the Grid: Enabling Scalable Virtual Organizations, *International Journal of High Performance Computing Applications*, Vol. 15, SAGE Publications, p. 200-222.

Foster, I., Kesselman, C. and Tuecke, S. (1995), The Nexus Approach to Integrating Multithreading and Communication, *IEEE*, Vol. 16, p. 1-13.

Foster, I., Vockler, J., Wilde, M. and Zhao, Y. (2002b), Chimera: A Virtual Data System for Representing, Querying, and Automating Data Derivation, in: *Proceedings of the 14th International Conference on Scientific and Statistical Database Management, 2002*, Edinburgh p. 37- 46.

Fox, G. (2003), Data and Metadata on the Semantic Grid, *Web Computing: Computing in Science and Engineering*, p. 76-78.

Fox, G., Gannon, D. and Thomas, M. (2002), Editorial: A Summary of Grid Computing Environments, *Concurrency Computation Practice Experience*, Vol. 14, John Wiley & Sons, Ltd, p. 1035–1044.

Fox, G., Pallickara, S., Pierce, M., and Gadgil, H. (2005), Building MessagingSubstrates for Web and Grid Applications, *Phil. Trans. R. Soc. A.*, Vol. 36, No. 3, p. 1757–1773.

geodise.org (2006), Geodise Project. Available at: www.geodise.org (accessed 20th June 2006).

Gero, J. S. (1994), Computational Models for Creative Design Processes, *Artificial Intelligence and creativity*, Vol. 64, No. 2, p. 269-280.

Ghanem, M., Guo, Y., Rowe, A. and Wendel, P. (2002), Grid-based Knowledge Discovery Services for High Throughput Informatics, in: *Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing* , ISBN:0-7695-1686-6, p. 416.

Gil, Y., Deelman, E., Blythe, J., Kesselman, C. and Tangmunarunkit, H. (2004), Artificial Intelligence and Grids: Workflow Planning and Beyond, *IEEE Intelligent Systems*, Vol. 40, p. 26-33.

globus.org (2006), Globus 4.0.4 Toolkit Project. Available at: www.globus.org (Accessed 25th January 2007).

Glover, F. (1989), Tabu Search-Part I, *ORSA Journal on Computing*, Vol. 1, no. 3, p. 190-206.

- Goble, C. and De Roure, D. (2002), The Grid: An Application of the Semantic Web, *SIGMOND Record*, Vol. 31, No. 4, p. 65-70.
- Goel, S. and Sobolewski, M. (2003), Trust and Security in Enterprise Grid Computing Environment, in: *Proceedings of the IASTED International Conference on Communication, Network and Information Security*, Vol. 23, April 407 2003, New York, p. 47-56.
- Goodyer, C. E., Berzins, M., Jimack, P. K. and Scales, L. E. A. (2005), Grid-Enabled Problem Solving Environment for Parallel Computational Engineering Design, *Advances in Engineering Software*, Vol. 37, No. 7, p. 439-449.
- Goteng, G., Tiwari, A. and Roy, R. (2008), Virtualisation of Grid Services for Multidisciplinary Optimisation, in: *Cranfield Multi-Strand Conference: Creating Wealth Through Research and Innovation*, , 6-7 May 2008, Cranfield University, UK, 6 pages.
- Goux, J., Kulkarni, S., Yoder, M. and Linderoth, J. (2001), Master-Worker: An Enabling Framework for Applications on the Computational Grid, *Cluster Computing*, Vol. 4, p. 63-70.
- Grace, P., Coulson, G., Blair, G, Mathy, L., Yeung, W. K., Cai, W., Duce, D. and Cooper, C. (2004), GRIDKIT: Pluggable Overlay Networks for Grid Computing, *FGCS*, Vol. 32, No. 91, p. 1463-1481.
- Grady, M. J., O'hare, G. M. P. and Donaghey, C. (2007), Delivering Adaptivity Through Context-Awareness, *Journal of Network and Computer Applications*, Vol. 30, p. 1007-1033.
- Grauer, M., Barth, T., and Thilo, F. (2004), Grid-based Computing for Multidisciplinary Analysis and Optimization, in: *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 30-31 Aug 2004, AIAA, p. 209-220.
- Hau, J., Lee, W. and Newhouse, S. (2003), Autonomous Service Adaptation in ICENI using Ontological Annotation, in: *Proceedings of the Fourth International Workshop on Grid Computing*, 17 Nov 2003, DC, USA, p. 10-17.
- Hayes, B. (2008), Cloud Computing, *News Technology-Communications of the ACM*, Vol. 51, No. 7, p. 9-11.
- He, Q. and Wang, L. (2007), A Hybrid Particle Swarm Optimization with a Feasibility-based Rule for Constrained Optimization, *Applied Mathematics and Computation*, Vol. 18, No. 6, p. 1407-1422.
- Hey, T. and Trefethen, A. (2003), e-Science and its Implications, *The Royal Society*, Vol. 36, No. 1, p. 1809-1825.
- Hey, T. and Trefethen, A. E. (2002), The UK e-Science Core Programme and the Grid, *Future Generation Computer Systems*, Vol. 18, p. 1017-1031.
- Hightower, J. and Borriello, G. (2001), Location Systems for Ubiquitous Computing, *IEEE Location Aware Computing*, Vol. 34, No. 8, P. 57-66.

Hwang, J. and Aravamudham, P. (2004), Middleware Services for P2P Computing in Wireless Grid Networks, *IEEE Internet Computing*, Vol. 8, No. 4, p. 40-46.

Iamnitchi, A. and Talia, D. (2005), P2P Computing and Interaction with Grids, *Future Generation Computer Systems*, Vol. 21, No. 3, p. 331-332.

Jackson, T., Austin, J., Fletcher, M. and Jessop, M. (2003), Delivering a Grid enabled Distributed Aircraft Maintenance Environment (DAME), *AHM*, 02-04 Sept. 2003, Nottingham, UK, p. 123-128.

Jie, L. and Jian-gang, Y. (2004), Holonic Approach for Resource Allocation in Agile Manufacturing Environment, *Concurrent Engineering: The Worldwide Engineering Grid*, Vol 57, No. 2, p. 75-80.

Johnston, W. E. (2003), Implementing Production Grids, in: Berman, F., Fox, G. C., and Hey, A. J. G., *Grid Computing: Making the Global Infrastructure a Reality*, Wiley Series, p. 118-167.

Jokela, T., Iivari, N., Matero, J., and Karukka, M. (2002), The Standard of User-Centered Design and the Standard Definition of Usability: Analyzing ISO 13407 against ISO 9241-11, in: *ACM Proceedings Series*, Vol. 46, p. 53-60.

Kacsuk, P., Goyeneche, A., Delaitre, T., Kiss, T., Farkas, Z. and Bocko, T. (2004), High-level Grid Application Environment to Use Legacy Codes as OGSA Grid Services, in: *Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing*, 8 Nov 2004, p. 428-435.

Kim, J. H., Lee, H. J., Kim, S. H., and Lee, J. O. (2008), A Problem Solving Environment Portal for Multidisciplinary Design Optimization, *Advances in Engineering Software*, Vol. 40, No. 8, p. 623-629.

Kosar, T. and Balman, M. (2009), A New Paradigm: Data-Aware Scheduling In Grid Computing, *Future Generation Computer Systems*, Vol. 25, No. 4, p. 406-413.

Kosar, T. and Livny, M. S. (2004), Making Data Placement a First Class Citizen in the Grid, in: *Proceedings of the 24th International Conference on Distributed Computing Systems*, 5-7 May 2004, USA, p. 54-62.

Kranzlmuller, D., Kurka, G., Hei-Nzleiter, P. and Volkert, J. (2003), Optimizations in the Grid Visualization Kernel, in: *Proceedings of the International Parallel and Distributed Processing Symposium*, 2002, p. 129-135.

Krauter, K., Buyya, R. and Maheswaran, M. (2002), A Taxonomy and Survey of Grid Resource Management Systems for Distributed Computing, *Software-Practice and Experience*, Vol. 32, p. 135-164.

Kumar, S., Dutta, K., and Mookerjee, V. (2009), Maximizing Business Value by Optimal Assignment of Jobs to Resources in Grid Computing, *European Journal of Operational Research*, Vol. 19, No. 4, p. 856-872.

Kurc, T., Catalyurek, U., Zhang, X., Saltz, J., Martino, R., Wheeler, M., Peszyńska, M., Sussman, A., Hansen, C., Sen, M., Seifoullaev, R., Stoffa, P., Torres-Verdin, C.

- and Parashar, M. (2005), A Simulation And Data Analysis System for Large-Scale, Data-Driven Oil Reservoir Simulation Studies, *Concurrency Computation and Practice Experience*, Vol. 17, p. 1441–1467.
- Lee, H. J., Lee, J. W., and Lee, J. O. (2009), Development of Web services-based Multidisciplinary Design Optimization framework, *Advances in Engineering Software*, Vol. 40, p. 176–183.
- Li, B. and Min, L. (2005), A Grid-based Architecture of Enterprise Total Cost Management Model, in: *Proceedings of the First International Conference on Semantics, Knowledge, and Grid*, 27-29 Nov 2005, China, p. 120-120.
- Li, Y. and Lu, Z. (2004), Ontology-based Universal Knowledge Grid: Enabling Knowledge Discovery and Integration on the Grid, in: *Proceedings of the 2004 IEEE International Conference on Services Computing*, 15-18 Sept. 2004, p. 557- 560.
- Lin, M. and Lin, Z. (2006), A Cost-Effective Critical Path Approach for Service Priority Selections in Grid Computing Economy, *Decision Support Systems*, Elsevier B.V., Vol. 42, p. 1628-1640.
- Litzkow, M. J, Livny, M., and Mutka, M. W. (1988), Condor: A Hunter of Idle Workstations, in: *IEEE 8th International Conference on Distributed Computing Systems*, 13-17 Jun 1988, p. 104-111.
- Liu, H., Jiang, L., Parashar, M., and Silver, D. (2005), Rule-based visualization in the Discover computational steering collaboration, *Future Generation Computer Systems*, Vol. 21, p. 53–59.
- Liu, H. and Orban, D. (2008), GridBatch: Cloud Computing for Large-Scale Data-Intensive Batch Applications, in: *Eighth IEEE International Symposium on Cluster Computing and the Grid*, 19-22 May 2008, p. 295-305.
- Luna, F., Nebro, A. J. and Alba, E. (2006), Observations in using Grid-enabled Technologies for Solving Multi-objective Optimization Problems, *Parallel Computing*, Vol. 32, No. 5-6, p. 377-393.
- Lynden, S., Mukherjee, A., Hume, A. C., Fernandes, A. A. A., Paton, N. W., Sakellariou, R., and Watson, P. (2009), The Design And Implementation of OGSA-DQP: A Service-Based Distributed Query Processor, *Future Generation Computer Systems*, Vol. 25, p. 224-236.
- Michalewicz, Z., Dasgupta, D., Riche, R. G. L., and Schoenauer, M. (1996), Evolutionary Algorithms for Constrained Engineering Problems, *Computers Industrial Engineering*, Vol. 30, No. 4, p. 851-870.
- Nebro, A. J., Alba, E. and Luna, F. (2007), Multi-Objective Optimization using Grid Computing, *Soft Computing*, Vol. 11, p. 531-540.
- Norton, A. and Rockwood, A. (2003), Enabling View-Dependent Progressive Volume Visualization on the Grid. *IEEE Graphic Applications for Grid Computing*, 22-31.

Ong, G. and Jiang, B. (2004), Experience and Lessons Learned from Enabling Applications for Users in National Grid Pilot Platform, *International Journal of Information Technology*, Vol. 11, No. 3, p. 9-15.

Ong, M., Ren, X., Allan, G., Kadirkamanathan, V., Thompson, H.A. and Fleming, P.J.. (2005), Decision Support System on the Grid, *International Journal of Knowledge-Based and Intelligent Engineering Systems*, Vol. 9, p. 315-326.

Papalambros, P. Y. (2002), The Optimization Paradigm in Engineering Design: Promises and Challenges, *Computer-Aided Design*, Elsevier, Vol. 34, p. 939-951.

Papazoglou, M. P. (2003), Service-Oriented Computing: Concepts, Characteristics and Directions, *Proceedings of the Fourth International Conference on Web Information Systems Engineering*, 10-12 Dec. 2003, Netherlands, p. 3-12.

Parashar, S. and Bloebaum, C. L. (2006), Multi-Objective Genetic Algorithm Concurrent Subspace Optimization (MOGACSSO) for Multidisciplinary Design, *47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 1 - 4 May 2006, Newport, Rhode Island, AIAA, p. 1.

Parashar, M., Klie, H., Catalyurek, U., Kurc, T., Bangerth, W., Matossian, V., Saltz, J. and Wheeler, M. F. (2005a), Application of Grid-Enabled Technologies for Solving Optimization Problems in Data-Driven Reservoir Studies, *Future Generation Computer Systems*, Vol. 21, p. 19-26.

Parashar, M., Matossian, V., Bangerth, W., Klie, H., Rott, B., Kurc, T., Catalyurek, U. J. and Wheeler, M. F. (2005b), Towards Dynamic Data-Driven Optimization of Oil Well Placement, *ICCS*, p. 656-663.

Parmee, I. C., Abraham, J., Shackelford, M., Rana, O. F. and Shaikhali, A. (2005), Towards Autonomus Evolutionary Design Systems via Grid-Based Technologies, in: *Proceedings of ASCE 2005 International Conference on Computing in Civil Engineering*, Cancun, Mexico, p. 12-19.

Pearlman, L., Welch, V., Foster, I., Kesselman, C. and Tueke, S. (2002), A Community Authorization Service for Group Collaboration, in: *Proceedings of the Third International Workshop on Policies for Distributed Systems and Networks*, p. 50-59.

Pouchard, L., Cinquini, L., Drach, B., Middleton, D., Bernholdt, D., Chanchio, K., Foster, I., Nefedova, A., Brown, D., Fox, P., Garcia, J., Strand, G., Williams, D., Chervenak, A., Kesselman, C., Shoshani, A. and Sim, A. (2003), An Ontology for Scientific Information in a Grid Environment: The Earth System Grid, in: *Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid*, Vol. 16, p. 626.

questionpro.com (2006), Available at: www.questionpro.com (Accessed 15th July 2006).

- Rajasekar, A., Wan, M. and Moore, R. (2002), MySRB & SRB - Components of a Data Grid, in: *Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing*, p. 301- 310.
- Rajasekar, A., Wan, M., Moore, R., Kremenek, G. and Guptil, T. (2003), Data Grids, Collections, and Grid Bricks, in: *Proceedings of the 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems Technologies*, 7-10 April 2003, CA, USA, p. 2-9.
- Ramakrishnan, R. (2008), Cloud Computing-Was Thomas Watson Right After All? *ICDE 2008*, 7-12 April 2008, p. 8-8.
- Ranganathan, K. and Foster, I. (2002), Decoupling Computation and data Scheduling in Distributed Data-Intensive Applications, in: *Proceedings of 11th IEEE International Symposium on High Performance Computing*, pp. 352-358.
- Rardin, R. L. and Uzroy, R. (2001), Experimental Evaluation of Heuristic Optimization Algorithms: A Tutorial, *Journal of Heuristics*, Vol. 7, p. 261-304.
- Ray, T. and Liew, K. M. (2002), A Swarm Metaphor For Multiobjective Design Optimization. Templeman, A. B. *Engineering Optimization*, Taylor and Francis, Vol. 34, p. 141-153.
- Rohl, P. J., Kolonay, R.M., Irani, R. K., Sobolewski, M., Kao, K. and Bailey, M. W. (2000), A Federated Intelligent Product Environment, *American Institute of Aeronautics and Astronautics*, AIAA-2000-4902, Sept. 6-8, 2000, p. 445-452.
- Roy, R. (1997), *Adaptive Search and the Preliminary Design of Gas Turbine Blade Cooling Systems*, (PhD Thesis), University of Plymouth, UK.
- Roy, R. and Mehnen, J. (2008), Dynamic Multi-Objective Optimisation for Machining Gradient Materials, *CIRP Annals-Manufacturing Technology*, Vol. 57, p. 429-432.
- Russell, R., Allen, G., Daus, G., Foster, I., Seidel, E., Novotny, J., Shalf, J. and Laszewski, G. V. (2002), The Astrophysics Simulation Collaboration: A Science Portal Enabling Community Software Development, *Cluster Computing*, Vol. 5, p. 297-304.
- Sacerdoti, F. D., Chandra, S. and Bhatia, K. (2004), Grid Systems Deployment & Management using Rocks, *Cluster 2004*, p. 337 - 345.
- Sasaki, D., Keane, A. J. and Shahpar, S. (2006), Multiobjective Evolutionary Optimization of a Compressor Stage using a Grid-Enabled Environment, in: *44th AIAA Aerospace Sciences Meeting and Exhibit*, 9 - 12 Jan, 2006, Nevada, USA. P. 444-453.
- Schikuta, E. and Weishaup, T. (2004), N2Grid: Neural Networks in the Grid, Vol. 2, 25-29 July 2004, p. 1409-1414.

- Schwidder, J., Talbott, T. and Myers, J. (2005), Bootstapping to a Semantic Grid, in: *2005 IEEE International Symposium on Cluster Computing and the Grid*, Vol. 1, 9-12 May 2005, p. 176-181.
- Shalf, J., Bethel, E. W. and Berkeley, L. (2003), Visualization Viewpoints: The Grid and Future Visualization System Architectures, *IEEE Computer Society*, p. 6-9.
- Silva, V. (2006), *Grid Computing for Developers*, (1st ed), Charles River Media - Programming Series, MA, USA.
- Slomiski, A. (2005), On Using BPEL Extensibility to Implement OGSF and WSRF Grid Workflows, *Concurrency and Computation*, Vol. 18, No. 10, p. 1229 - 1241
- Smith, C. and Lumb, I. (2001), The Open Grid Services Architecture, in: Abbas, A. (editor), *Grid Computing: A Practical Guide to Technology and Applications*, Charles River Media, USA, p. 159-187.
- Sobieszcanski-Sobieski, J. and Haftka, R. T. (1997), Multidisciplinary Aerospace Design Optimization: Survey of Recent Developments, *American Institute of Aeronautics and Astronautics*, Vol. 14, No. 1, p. 1-23.
- Sobolewski, M. (2007). Metacomputing with Federated Method Invocation, *CE2007*, p. 23-30.
- Sobolewski, M. and Kolonay, R. M. (2006), Federated Grid Computing with Interactive Service-oriented Programing, *CE2006*, Vol. 14, No. 1, p. 56-66.
- Sobolewski, M., Soorianarayanan, S. and Malladi-Venkata, R. K. (2003), Service-Oriented File Sharing, in: *Proceedings of the 2nd IASTED International Conference, Communications, Internet and Information Technology*, Vol. 408, p. 370-373.
- Soldatos, J., Pandis, I., Stamatis, K., Polymenakos, L. and Crowley, J. L. (2005), Agent Based Middleware Infrastructure for Autonomous Context-Aware Ubiquitous Computing Services, *Computer Communications*, Vol. 30, p. 577-591.
- Song, W., Ong, Y. S., Ng, H. K., Keane, A., Cox, S. and Lee, B. S. (2004), A Service-Oriented Approach for Aerodynamic Shape Optimisation across Institutional Boundaries, in: *8th International Conference on Control, Automation, Robotics and Vision*, , Vol. 3, 6-9 Dec. 2004, Kunming, China, p. 2274- 2279.
- Soorianayanan, S. and Sobolewski, M. (2004), Monitoring Federated Services in CE Grids, *Concurrent Engineering Journal*, p.
- Sprott, D. and Wilkes, L. (2004), Understanding Service-Oriented Architecture, *The Architecture Journal*, p. 205-216.
- Stevens, R., Woodward, P., Defanti, T., and Catlett, C. (1997), From the I-WAY to the National Technology Grid., *Communications of ACM*, Vol. 40 No. 11, p. 51-60.
- Stockinger, H., Samar, A., Holtman, K., Allcock, B., Foster, I. and Stierney, B. (2002), File and Object Replication in Data Grids, *Cluster Computing*, Kluwer Academic Publishers, Neitherlands, p. 305-314.

- Sudholt, W., Baldrige, K. K., Abramson, D., Enticott, C., Garic, S., Kondric, C. and Nguyen, D. (2005), Application of Grid Computing to Parameter Sweeps and Optimizations in Molecular Modeling, *Future Generation Computer Systems*, Vol. 21, p. 27-35.
- Suh, N. P. (1995), Designing-in of Quality through Axiomatic Design, *IEEE Transactions on Reliability*, Vol. 44, No. 2, p. 256-264.
- Sundaresan, S., Ishii, K., and Houser, D. R. (1993), A Robust Optimization Procedure With Variations on Design Variables and Constraints, *Advances in Design Automation ASME*, Vol. 24, No. 2, p. 101-117.
- Surridge, M., Taylor, S., De Roure, D., and Zaluska, E. (2005), Experiences with GRIA – Industrial applications on a Web Services Grid, in: *Proceedings of the First International Conference on e-Science and Grid Computing*, DC, USA, p. 98-105.
- Sushil, J. L. (1993), *Genetic Algorithms as a Computational Tool for Design*. (PhD Thesis). Indiana University, USA.
- Talia, D. (2002), The Open Grid Services Architecture: Where The Grid Meets The Web, *IEEE Internet Computing*, Vol. 6, p. 67-71.
- Tezuka, S., Murata, H., Tanaka, S. and Yumae, S. (2005), Monte Carlo Grid for Financial Risk Management, *Future Generation Computer Systems*, Vol. 21, p. 811–821.
- Tiwari, A. (2001), *Evolutionary Computing Techniques for Handling Variable Interaction in Engineering Design Optimisation* (PhD Thesis), Cranfield University, UK.
- Tripathi, A., Wolfe, R., Koneru, S. and Attia, Z. (1992), Management of Persistent Objects in the Nexus Distributed System, *IEEE*, Vol. 20, p. 100-1004.
- Vazhkudai, S. and Schopf, J. M. (2002), Predicting Sporadic Grid Data Transfers, in: *Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing*, 2002. p. 188.
- Venugopal, S., Buyya, R. and Winton, L. (2004), A Grid Service Broker for Scheduling Distributed Data-Oriented Applications on Global Grids, *Practice & Concurrency*, Vol. 18, No. 6, p.685-699.
- Walsh, K. R. (2003), Analyzing the Application ASP Concept: Technologies, Economies and Strategies, *Communications of the ACM* , Vol. 46, No. 8, p. 103-107.
- Wan, M., Rajasekar, A., Moore, R. and Andrews, P. (2003), A Simple Mass Storage System for the SRB Data Grid, in: *Proceedings of the 20th IEEE /11th NASA Goddard Conference on Mass Storage Systems and Technologies*, 7-10 April 2003, p. 20-25.
- Wang, D., Wang, G. G. and Naterer G. F. (2005), Collaboration Pursuing Method for MDO Problems, in: *1st AIAA Multidisciplinary Design Optimization Specialists Conference*, Vol. 45, AIAA.

- Wang, F., Wu, S., Helian, N., Parker, A., Guo, Y., Deng, Y. and Khare, V. (2007), Grid-oriented Storage: A Single-Image, Cross-Domain, High-Bandwidth Architecture, *IEEE Transaction on Computers*, Vol.56, No.4, p. 474-487.
- Wang, L. and Jie, W. (2009), Towards Supporting Multiple Virtual Private Computing Environments on Computational Grids, *Advances in Engineering Software*, Vol. 40, p. 239–245.
- Weiss, A. (2007), Computing in the Clouds, *ACM*, Vol. 11, No. 6, p. 16-25.
- Welch, V., Siebenlist, F., Foster, I., Bresnahan, J., Czajkowski, K., Gawor, J., Kesselman, C., Meder, S., Pearlman, L. and Tueke, S. (2003), Security for Grid Services, in: *Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing*, 22-24 June 2003, IL, USA, p. 48- 57.
- Wells, A. J. (2008), *Grid Application Systems Design* (2nd ed), Averbach Publications-Taylor and Francis Group, USA.
- Wolstencroft, K., Alper, P. and Hull, D. (2007), The myGrid Ontology: Bioinformatics Service Discovery, in: *Int. Journal of Bioinformatics Research and Applications*, Vol. 3, No. 3, p. 303-325.
- Wu, B. Y., Chi, C. H., Chen, Z., Gu, M., and Sun, J. G. (2009), Workflow-Based Resource Allocation to Optimize Overall Performance of Composite Services, *Future Generation Computer Systems*, Vol. 25, p. 199-212.
- Xue, G., Song, W., Keane, A. J. and Cox, S. J. (2004), Developing Services for Design Optimisation on the Grid, *Proceedings of the 2004 IEEE International Conference on Services Computing*, 15-18 Sept. 2004, UK, p. 391- 398.
- Yu, J. and Buyya, R. (2006), A Taxonomy of Workflow Management Systems for Grid Computing, *Journal of Grid Computing*, Vol. 3, p. 171–200.
- Zhao, S. and Sobolewski, M. (2001), Context Model Sharing in the FIPER Environment, in: *Proceedings of the 8th Int. Conf. on Concurrent Engineering*, p. 12-20.
- Zhuge, H. (2004), China's E-Science Knowledge Grid Environment, *IEEE Intelligent Systems*, p. 13-17.
- Zhuge, H. and Liu, J. (2005), A Fuzzy Collaborative Assessment Approach for Knowledge Grid, *Future Generation Computer Systems*, Vol. 20, p.101-111.
- Zitzler, E. (1999), *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications* (PhD Thesis), Swiss Federal Institute of Technology, Zurich.
- Zitzler, E. and Thiele, L. (1998), *An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach*, Report Number 43, TIK Report.

Appendix-I

Questionnaires for Industry Survey

To gather data during the industry survey, questionnaire was prepared. The questionnaires were in two parts-the hard copy (paper) questionnaire and the online one. The hard copy questionnaire used is shown below. The first part of it is the introductory letter and the followed by the questionnaire. Prior to this, contacts were made with the respondents before embarking on the visits.

PhD Research: Grid Computing-Enabled Engineering Design Optimisation within an Extended Enterprise

Researcher:

Gokop Goteng, Affiliation: Cranfield University, UK

Supervisors:

Dr. Ashutosh Tiwari, Affiliation: Cranfield University, UK

Prof. Rajkumar Roy, Affiliation: Cranfield University, UK

Dear Sir/Madam,

INDUSTRIAL SURVEY: QUESTIONNAIRE

The Decision Engineering Centre of Cranfield University, UK is working on the above PhD research topic for the next 3 years (2006-2009) and wishes to solicit for your support and time in filling this questionnaire. The aim of the questionnaire is to identify companies that are using or intend to use Grid Computing for certain aspects of their operations and also to know the benefits and problems associated with Grid implementation issues.

We assure you of absolute confidentiality in dealing with the information you may provide for us. We also promise to share the results/findings of the research with your company by submitting a copy of the thesis to you on completion of the research.

We hope you will find the questionnaire easy to follow. Do not hesitate to ask for clarification where any question(s) is/are not clear enough.

Yours sincerely,

Gokop Goteng

QUESTIONNAIRE

Name:							
Name of Organisation:							
Department:				Position			
1. Do you Know about Grid Computing? (Yes/No)			2. Does your Company uses Grid Computing or intends to use it? (Yes/No)				
If No in (2), What do you think is the reason?							
3. If Yes in (2), why do you think your company wants to implement Grid Computing? (tick one or more)							
Competitive Advantage		Cost Reduction		Reduce Time to Market		Others (Specify)	
4. If Yes in (2), Which area of application does your company uses or intends to use Grid for? (tick one or more)							
Computation		Data Intensive		Visualisation		Collaboration	
						Sharing Resources	
Multidisciplinary Problem Solving Environment		Knowledge Repository		Others (Specify)			
5. If Yes in (2), in what discipline does your company uses or intends to use Grid Computing? (tick one or more)							
Engineering Design Optimisation		General Engineering		Physics		Bioinformatics	
						Pharmaceuticals	
Medicine		General Science		Service-Oriented Applications		Oil and Gas	
						Others (Specify)	
6. If Yes in (2), Which of the following Grid Middleware does your Company/Organisation uses or intends to use?							
Globus		Condor		Nimrod-G		GridBus	
						Others (Specify)	
7. Why do you think your company/organisation chooses to use the above Grid Middleware?							
8. If Yes in (2), Which of the following Grid Problem Solving Environment (PSE) your company/organisation uses or intends to use?							
Globus		Geodise		FIPER		SORCER	
Matlab		DAME		Gridbus		Others (Specify)	
9. Why do you think your company/organisation chooses to use or intends to use the above PSE?							
10. Do you think that Grid Security is a threat to sharing resources and collaboration among companies and organisations? (Yes/No)							
11. If Yes in (10), What do you think are the security threats?							
12. Which of the following application do you think Grid is suitable for? (tick one or more)							
Scientific		Engineering		Business		Data mining	
						Others (Specify)	
13. Please can you explain briefly why you choose the above application area (s) as suitable for Grid?							

There was also a telephone interview conducted with Microsoft Technology Officer in UK. Telephone interview was scheduled due to time constraint on the part of the respondent. The interview was brief, but very useful. The questionnaire used is shown below.

Microsoft Interview Questions (10 am: 12/01/2007)

1. Do you think the enthusiasm surrounding the emerging grid technology will deliver business benefits to companies?
2. One of the challenges in Grid is developing a stable software development environment (SDE) for grid application programmers just like .NET is used by windows and web-based developers. Is your company looking at developing a platform for grid application developers?
3. How can you rate the awareness of grid computing outside research and academic communities?
4. How do you see the issue of security in using grid for collaboration
5. What do you think are the challenges in grid implementation?
6. What is the Microsoft Compute Cluster Server supposed to deliver?
7. Will it be more intuitive and have better graphical user interface than Linux?
8. Do you think that the academia and researchers will be willing to pay for the MS Compute Cluster and abandon Linux? Besides they are already used to it?
9. Why did it take Microsoft this long to produce its first version of the compute cluster?
10. Any other comments?

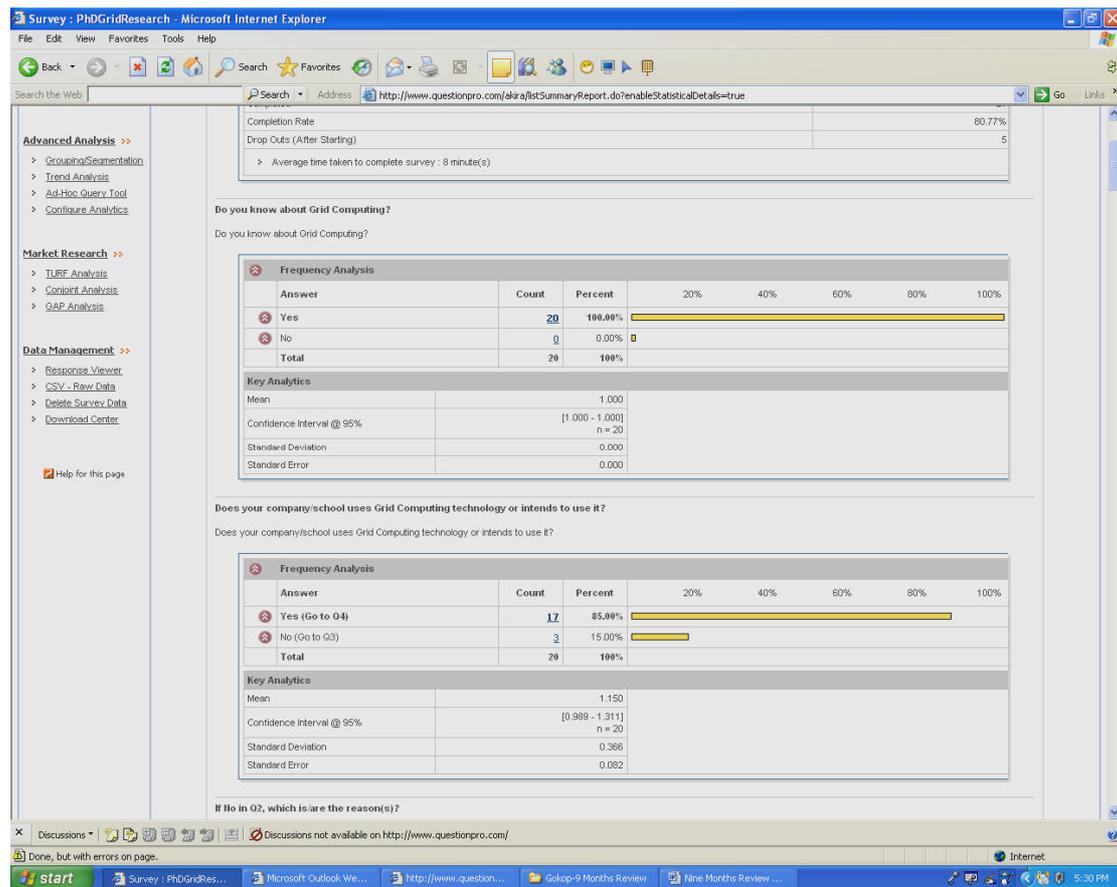
Thank you

Gokop Goteng

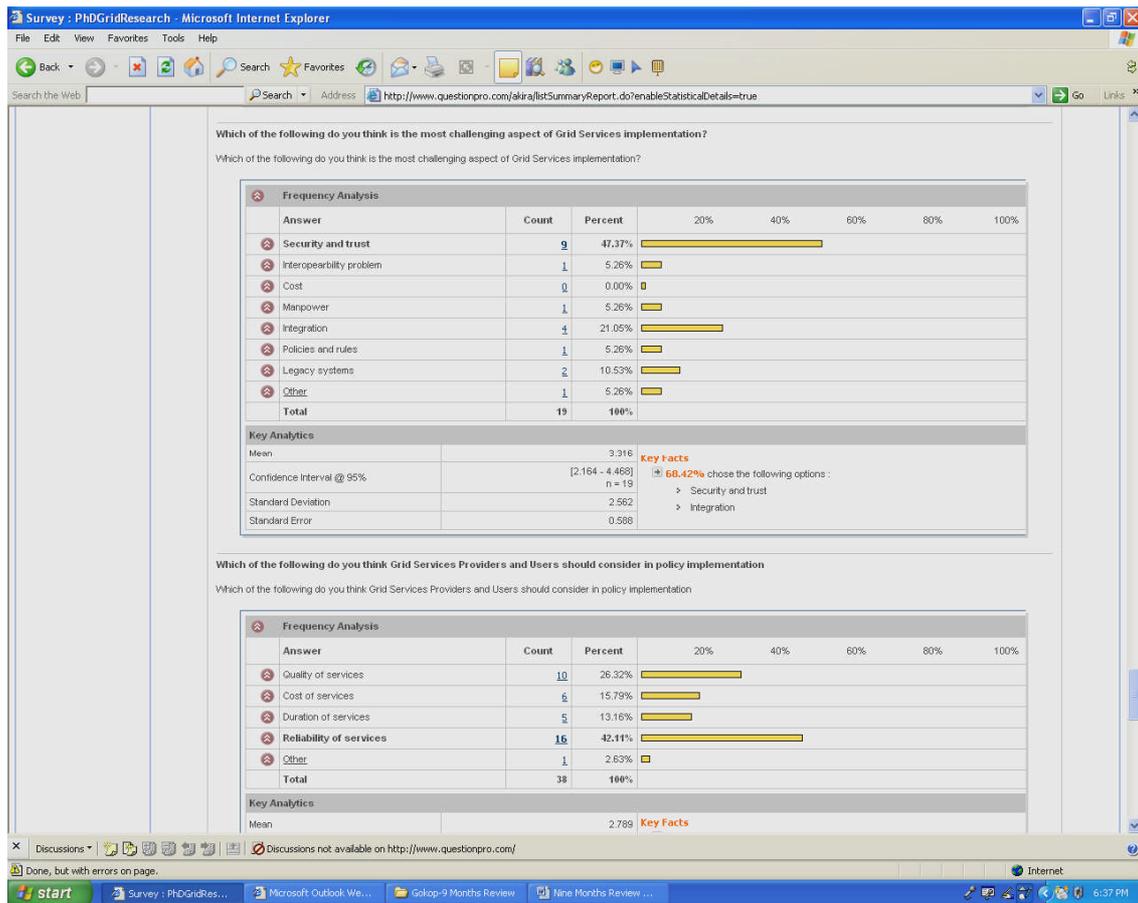
Online Questionnaire using QuestionPro Package

An online questionnaire was used to gather data. This proved to be very useful as the researcher was able to reach to respondents from different countries of the world. Some screenshots of the online questionnaire are as below. An e-mail was first sent with the link to the questionnaire.

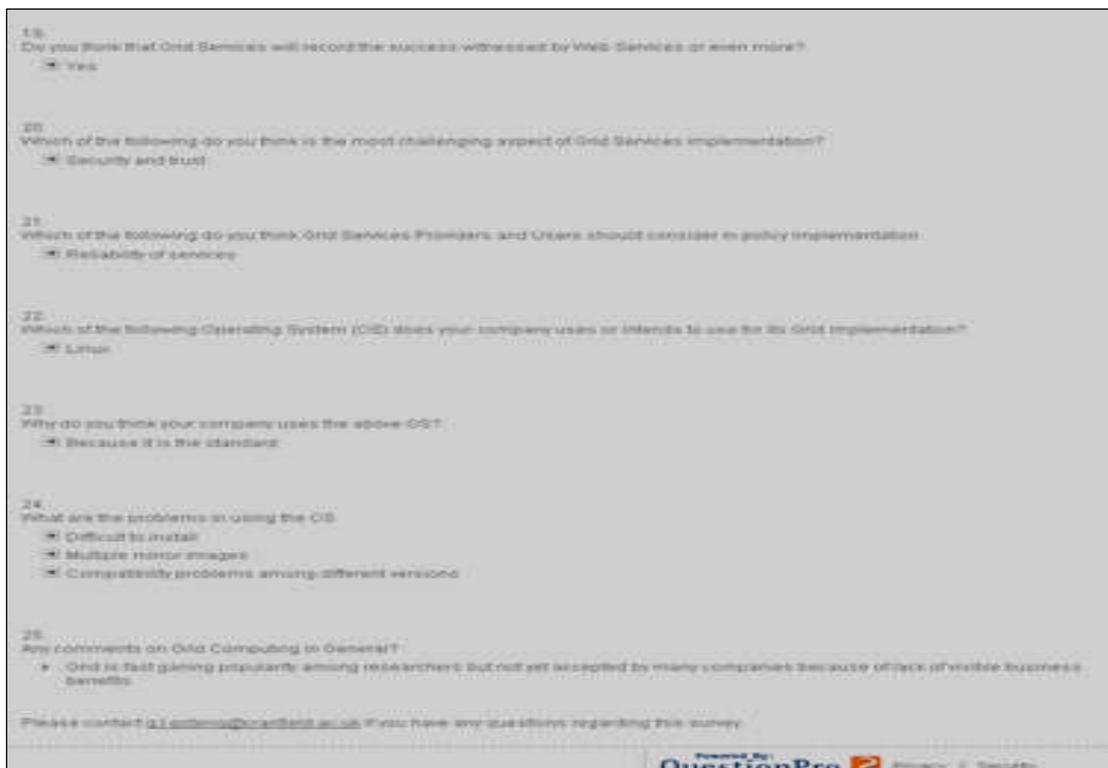
The questions in this screen try to know how many respondents even know about grid computing and perhaps are using it.



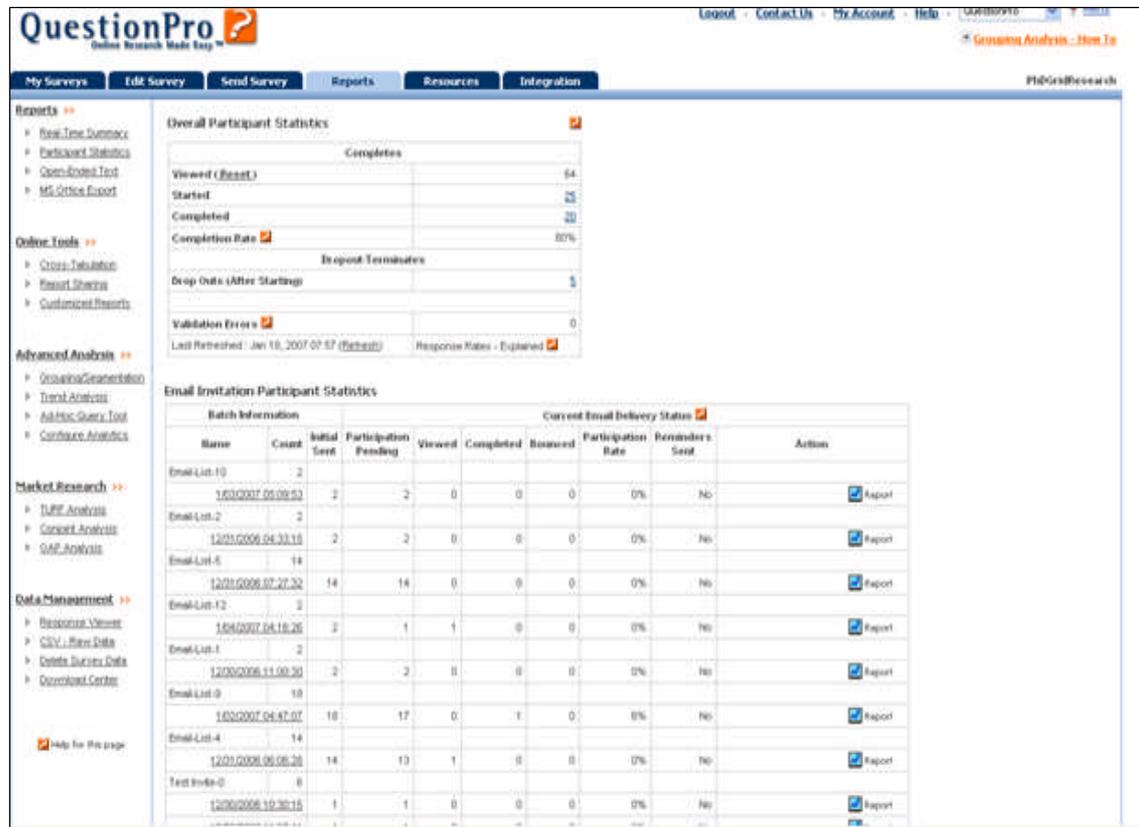
This screenshot takes a look at the most challenging aspect of grid implementation and adoption by companies and researchers.



This screen shot is trying to ask how successful grid technology may be among companies.



This screenshot shows the summary of how many people responded to the questions, how many viewed it and the dropped out and how many started filling it and then dropped out.



Respondents were given questionnaire to fill after the demonstration of the prototype. Sample of the questionnaire is below.

Validation Questionnaire

Please fill in the following questionnaire after this workshop for the purpose of validating the DECGrid prototype.

1. Do you think the system achieved the aim and objectives of the research and for what it is intended to do? Please make comments

2. What are the limitations of the prototype?

3. Suggest any improvement that can be done to make the system better.

4. Is the system useful? And does it produce good results compared with the ones in literature?

5. Does the system allow users to perform MODO in distributed environment?

6. Does the service provide essential components to support a designer for MODO?

7. Is there any contribution to knowledge by using this system to perform optimisation?

Name/Signature and Date _____

Appendix-II

Programming Codes to Run the NSGA-II for the Case Studies

The programming codes for the quantitative models were written in c while the qualitative is written in c++ programming language. The grid services were written in a combination of java, HTML, XML, PHP and javascript languages.

Definition and Declarations of the Problem Variables and Constants for the 3 Case Studies

```
# include <stdio.h>
# include <stdlib.h>
# include <math.h>
# include "global.h"
# include "rand.h"
#define true 1
#define false 0
typedef int boolean;
char qua_value;
FILE *pfl;
population *pop;
int pausing;
# define turbineblade
# define weldedbeam
# define designofmanufacturinglayout
```

Turbine Blade Cooling System C Codes

```
#ifdef turbineblade
void test_problem (double *xreal, double *xbin, int
**gene, double *obj, double *constr)
{
double geom, Cdr, Fhc, Tc1, dth, kw, Rp, Rs, df, Cdf, Ff,
Rpf;
double results[20];
boolean flag;
int geometry = 0;
geom = xreal[0];
Cdr = xreal[1];
Fhc = xreal[2];
Tc1 = xreal[3];
dth = xreal[4];
kw = xreal[5];
Rp = xreal[6];
Rs = xreal[7];
```

```
df = xreal[8];
Cdf = xreal[9];
Ff = xreal[10];
Rpf = xreal[11];
geometry = floor( geom + 0.5 );
int runs = 0;
int decpt = 0;
    int Nb = 0;
    Nb = 78;
    double Whpc = 0;
    Whpc = 84.85;
    double Wcr1 = 0;
    Wcr1 = 0.003*(Whpc/Nb);
    double Twg = 0;
    if ( geometry == 3 )
        Twg = 1050;
    else
        Twg = 1250;
    double Tc = Tc1;
    double Tg = 0;
    Tg = 1500.0;
    double hg = 0;
    hg = 3000;
    double Twc = 0;
    Twc = Twg - ((2*dth*hg*Rs*(Tg-Twg))/(kw*(Rs+1)));
    double hcr1 = 0;
    hcr1 = hg * Rs * (Tg-Twg)/(Twc-Tc);
    double hcr2 = hcr1;
    double k = 0;
    double u = 0;
    double F = 0.01855;
    double FF = 0;
    double Acr = 0;
    double Pc3 = 460000.0;
    double Pc1 = 0;
    double Pc2 = 0;
    double y = 1.36;
    double R = 287.0;
    double Wcr2 = Wcr1;
    double Twgold = Twg;
    double Twgnew = Twg;
    double Scr = 0;
    double H1 = 0;
    double lr = 0;
    double H2 = 0;
    double H3 = 0;
    double Cp = 993.0;
    int count1 = 0;
    int count2 = 0;
    int count3 = 0;
    int count4 = 0;
```

```

do {
    do {
        hcr1 = hcr2;
        Wcr1 = Wcr2;
        k = ((2.978E-03) * pow(Tc,0.5)) / (1 + (240.0/Tc));
        u = ((1.488E-06) * pow(Tc,1.5)) / (Tc+110.4);
        FF = F * Fhc;Acr =
pow(FF*(k/pow(u,0.8))*(pow(Wcr1,0.8)/hcr1),(1.0/0.9));

        while (Acr > (2.75E-05)) {

            Wcr1 = 0.99*Wcr1;
            Acr =
pow(FF*(k/pow(u,0.8))*(pow(Wcr1,0.8)/hcr1),(1.0/0.9));
        }
        Pc1 = Pc3 * Rp;
        Pc2 = Pc1 - Rpf * (Pc1-Pc3);
        Wcr2 =
((Acr*Cdr*Pc1)/pow(Tc1,0.5))*pow(((2*y)/(R*(y-
1))*(pow(Pc1/Pc2,(-2/y))-pow(Pc1/Pc2,((-1-y)/y)))),0.5);
        hcr2 =
FF*(k/pow(u,0.8))*(pow(Wcr2,0.8))/(pow(Acr,0.9));
        count1 = count1 + 1;
    } while ( ((fabs(hcr2-hcr1) > 10) || (hcr2 <
100) || (hcr2 > 4000)) && (count1 < 100) );
    do {
        Scr = 3.545*pow(Acr,0.5);
        H1 = hcr2/(hg*Rs);
        lr = 0.0406;
        H2 = (hcr2*Scr*lr)/(2*Wcr2*Cp);
        H3 = (kw/(dth*hg))*0.5*(1+(1/Rs));
        Twg = ((1+H2-(H1*H2)/(H1+H3))*Tg+(H1-
(H1*H1)/(H1+H3))*Tc1)/(1+H2-
(H1*H2)/(H1+H3)+(H1*H3)/(H1+H3));
        if ( (Twg < 1000.0) || (Twg > 1500.0) ) Wcr2 = Wcr2 *
1.01;

        count2 = count2 + 1;
    } while ( ((Twg < 1000.0) || (Twg > 1500.0))
&& (count2 < 100) );
    Twgold = Twgnew;
    Twgnew = Twg;
    Tc = (H2/H1)*(Tg-Twg)+Tc1;
    count3 = count3 + 1;
} while ( (fabs(Twgold-Twgnew) > 1) && (count3 < 100) );
Pc1 = Pc3 * Rp;
Pc2 = Pc1 - Rpf*(Pc1-Pc3);
double Af = 0;
int Nf = 30;
Af = Nf*0.25*3.142857*pow(df,2);
H1 = hcr2/(hg*Rs);
lr = 0.0406;

```

```

H2 = (hcr2*Scr*lr)/(2*Wcr2*Cp);
H3 = (kw/(dth*hg))*0.5*(1+(1/Rs));
Twg = ((1+H2-(H1*H2)/(H1+H3))*Tg+(H1-
(H1*H1)/(H1+H3))*Tc1)/(1+H2-
(H1*H2)/(H1+H3)+(H1*H3)/(H1+H3));
double Tc2 = 0;
Tc2 = (H2/H1)*(Tg-Twg)+Tc1;
double Wcf = 0;
Wcf = ((Af*Cdf*Pc2)/pow(Tc2,0.5))*pow((2*y/(R*(y-
1)))*(pow(Pc2/Pc3,-2/y)-pow(Pc2/Pc3,(-1-y)/y)),0.5);
H1 = hcr2/(hg*Rs);
lr = 0.0406;
H2 = (hcr2*Scr*lr)/(2*Wcr2*Cp);
H3 = (kw/(dth*hg))*0.5*(1+(1/Rs));
Twg = ((1+H2-(H1*H2)/(H1+H3))*Tg+(H1-
(H1*H1)/(H1+H3))*Tc1)/(1+H2-
(H1*H2)/(H1+H3)+(H1*H3)/(H1+H3));
Tc2 = (H2/H1)*(Tg-Twg)+Tc1;
Af = Nf*0.25*3.142857*pow(df,2);
Pc1 = Pc3 * Rp;
Pc2 = Pc1 - Rpf*(Pc1-Pc3);
H1 = hcr2/(hg*Rs);
lr = 0.0406;
H2 = (hcr2*Scr*lr)/(2*Wcr2*Cp);
H3 = (kw/(dth*hg))*0.5*(1+(1/Rs));
Twg = ((1+H2-(H1*H2)/(H1+H3))*Tg+(H1-
(H1*H1)/(H1+H3))*Tc1)/(1+H2-
(H1*H2)/(H1+H3)+(H1*H3)/(H1+H3));
Tc2 = (H2/H1)*(Tg-Twg)+Tc1;
Wcf = ((Af*Cdf*Pc2)/pow(Tc2,0.5))*pow((2*y/(R*(y-
1)))*(pow(Pc2/Pc3,-2/y)-pow(Pc2/Pc3,(-1-y)/y)),0.5);
double Xf = 10.0;
double dthf = 0;
dthf = dth/2.0;
double Mach = 0.6;
double Tc31 = Tc2;
double Tc31new = Tc31;
double Tc31old = Tc31;
double FF1 = 0;
double hcf = 0;
double H11 = 0;
double Scf = 0;
double lf = 0;
double H22 = 0;
double H33 = 0;
double Twfm = 0;

do {
Tc = (Tc2 + Tc31)/2.0;
k = ((2.978E-03) * pow(Tc,0.5)) / (1 +
(240.0/Tc));

```

```

        u = ((1.488E-06) * pow(Tc,1.5)) / (Tc+110.4);
        FF1 = Ff*F;
        hcf =
FF1*(k/pow(u,0.8))*(pow(Wcf,0.8)/pow(Af,0.9));
        H11 = hcf/(hg*Rs);
        Scf = Nf*3.142857*df;
        lf = 5.0*df;
        H22 = (hcf*Scf*lf) / (2*Wcf*Cp);
        H33 = (kw/(dthf*hg))*0.5*(1+1/Rs);
        Twfm = ((1+H22 - (H11*H22)/(H11+H33))*Tg+(H11-
(H11*H11)/(H11+H33))*Tc2)/(1+H22-
(H11*H22)/(H11+H33)+(H11*H33)/(H11+H33));
        Tc31 = ((2*H22)/H11)*(Tg-Twfm)+Tc2;
        Tc31old = Tc31new;
        Tc31new = Tc31;
        count4 = count4 + 1;
    } while ( (fabs(Tc31new-Tc31old) > 2.0) &&
(count4 < 100) );
    double tg = 0;
    tg = Tg/(1.0+((y-1)/2.0)*Mach*Mach);
    double RWA = 0;
    RWA = Mach*Pc3*pow(y/(R*tg),0.5);
    double Ef = 0;
    Ef = 0.66-
0.0092*pow(RWA*((Af*Cdf)/Wcf)*Xf,0.8)*pow(Tg/Tc31,0.6);
    double Tf = 0;
    Tf = Tg-Ef*(Tg-Tc31);
    H1 = hcr2/hg;
    H2 = (hcf*Scf*lf) / (2*Wcf*Cp);
    H3 = kw/(dth*hg);
    double Twf = 0;
    Twf = ((1+H2-(H1*H2)/(H1+H3))*Tf+(H1-
(H1*H1)/(H1+H3))*Tc1)/(1+H2-
(H1*H2)/(H1+H3)+(H1*H3)/(H1+H3));
    flag = true;
    if ( geometry == 1 ) {
        if ( (Cdr < 0.60) || (Cdr > 0.75) ) flag =
false;
        if ( (Fhc < 1.0) || (Fhc > 1.6) ) flag =
false;
    }
    if ( geometry == 2 ) {
        if ( (Cdr < 0.40) || (Cdr > 0.60) ) flag =
false;
        if ( (Fhc < 1.3) || (Fhc > 3.0) ) flag =
false;
    }
    if ( geometry == 3 ) {

```

```

        if ( (Cdr < 0.2) || (Cdr > 0.4 ) ) flag =
false;
        if ( (Fhc < 1.8) || (Fhc > 3.2 ) ) flag =
false;
    }

    if ( flag == true ) {
        results[0] = Wcr2;
        results[1] = Twg;
        results[2] = Wcf;
        results[3] = Twf;
    }
    else {
        results[0] = 1000000;
        results[1] = 1000000;
        results[2] = 1000000;
        results[3] = 1000000;
    }

obj[0] = results[0];
obj[1] = results[1];
obj[2] = results[2];
obj[3] = results[3];
constr[0] = results[1]-1200.0;
constr[1] = 1300.0-results[1];
constr[2] = 1300.0-results[3];
constr[3] = (results[0]/results[2])-0.8;
return;
}
#endif

```

Welded Beam Problem C Codes

```

#ifdef weldedbeam
void test_problem (double *xreal, double *xbin, int
**gene, double *obj, double *constr)
{

double F, dt, r, r1, r2, PC;
double b, t, h, l, l1, ht1, c1, c2, c3, c4, c5, c6, c7,
c8, c9, c10, c11, c12, c13;

    /*Add welded beam model below(This section was generated
by the math model building services*/
F= 6000;
b= xreal[0];
t= xreal[1];
h= xreal[2];

```

```

l= xreal[3];
l1= pow(l,2);
ht1= pow((h+t),2);
dt= 2.1952/(pow(t,3)*b);
r1= F/(pow((2*h*1),0.5));
c1=1.10471*pow(h,2)*1;
c2=0.04811*t*b;
c3=14.0+l;
c4=pow(r1,2);
c5=0.25*(l1+ht1);
c6=pow(c5,0.5);
c7=F*(14.0+(0.5*1));
c8=(l1/12)+(0.25*ht1);
c9=0.707*h*1;
c10=2*c9*c8;
c11=c7*c6;
r2=c11/c10;
c12=pow(r2,2);
c13=l*r1*r2;
dt= 504000/(pow(t,2)*b);
r=pow((c4+c12+(c13/c6)),0.5);
PC= 64746.022*(1-(0.0282346*t))*(t*pow(b,3));
obj[0]=c1+(c2*c3);
obj[1]= dt/5000000;
constr[0]= 13600-r;
constr[1]= 30000-dt;
constr[2]= b-h;
constr[3]= PC-F;
}
#endif

```

Design of a Manufacturing Layout C Codes

```

#ifdef designofmanufacturinglayout
void test_problem (double *xreal, double *xbin, int
**gene, double *obj, double *constr)
{
double C1, C2, C3, C4, C5, C6, C7;
FILE *file;
int numbers[30];
int i,j;
file = fopen("/var/www/html/qualitative_ratings.txt",
"r");
if(file==NULL) {
printf("Error: can't open file.\n");
}
else {
printf("File opened successfully.\n");
}
}
#endif

```

```

    i = 0 ;
    while(!feof(file)) { /*
/* loop through and store the numbers into the array */
    fscanf(file, "%d", &numbers[i]);
    i++;
    }

/* loop through and store the numbers into the array */
    for (i=0; i<popsize; i++) {
        fscanf(file, "%d", &numbers[i]);
    }
    printf("Number of numbers read: %d\n\n", i);
    printf("The numbers are:\n");
    }
    for(j=0 ; j<12 ; j++) {
        printf("%d\n", numbers[j]);
    }
fclose(file);
char cqv;
pfl= fopen("qualitative_ratings.txt","r");

if(!pfl)

{

printf("ERROR: Problem in file opening!\n");

}
else
{
cqv=fgets(qua_value, 6, pfl);
doubleqv=atof(cqv);*/
for (i=0; i<popsize; i++) {
obj[1]=numbers[i];
printf("%d\n", numbers[i]);
}
for (i=0; i<popsize; i++)
{

    for (j=0; j<nobj; j++)
    {
pop->ind[i].obj[1]=numbers[i];
printf("%d\n", numbers[i]);

    }

}
}
C1=xreal[0]*(2.2-xreal[1]);
C2=2*xreal[1]*xreal[2];
C3=xreal[3]*xreal[4];
C4=2*(3.6-(xreal[0]+xreal[6]))*xreal[5];
C5=xreal[6]*xreal[7];

```

```

C6=xreal[1]*(3.6-(xreal[2]+xreal[3]));
C7=((3.6-(xreal[0]+xreal[6]))*(2.2-
(xreal[1]+xreal[5])))+(xreal[6]*(2.2-
(xreal[1]+xreal[7])));
obj[0]=(C1+C2+C3+C4+C5+C6+C7)/(1+xreal[8]);

constr[0]=3.6-(xreal[0]+xreal[2]+xreal[3]+xreal[6]);
constr[1]=2.2-(xreal[1]+xreal[4]+xreal[5]+xreal[7]);
fclose(pf1);
    return 0;
}
printf("\nWaite to continue . . .");
    system("PAUSE");
}
}
#endif

```

Math Model Building Service Java Codes

```

import java.rmi.RemoteException;

import org.globus.wsrp.Resource;
import org.globus.wsrp.ResourceProperties;
import org.globus.wsrp.ResourceProperty;
import org.globus.wsrp.ResourcePropertySet;
import org.globus.wsrp.impl.ReflectionResourceProperty;
import org.globus.wsrp.impl.SimpleResourcePropertySet;

public class MODOService implements Resource,
ResourceProperties {

    /* Resource Property set */
    private ResourcePropertySet propSet;

    /* Resource properties */
    private string variablename;
    private String leftoperaor;
    private String rightoperaor;
    private String outputvariable;
    private String equationexpression;
    private String mathmodel;
    private int numberofiterations;
    private int noOfExpressions;

    /* Constructor. Initializes RPs */
    public MODOService() throws RemoteException {
        /* Create RP set */
        this.propSet = new SimpleResourcePropertySet(
            MODONames.RESOURCE_PROPERTIES);

        /* Initialize the RP's */
        try {
            ResourceProperty variableRP = new
            ReflectionResourceProperty(

```

```

                                MODOQNames.RP_Variable,
"Value", this);
                                this.propSet.add(variableRP);
                                setValue(0);
                                throw new RuntimeException(e.getMessage());
        }
    }

    /* Get/Setters for the RPs */

    public int getVariable() {
        return variable;
    }

    public void setVariable(string variable) {
        this.variable = variable;
    }

    public String getLeftOperator() {
        return leftOperator;
    }

    public void setLeftOperator(String leftOperator() {
        this.leftOperator = leftOperator;
    }
}

public void setRightOperator(String RightOperator() {
    this.rightOperator = rightOperator;
}

public void setOutputExpression(String OutputExpression() {
    this.outputExpression = outputExpression;
}

/* Remotely-accessible operations */

public GeneradMathModelResponse generate(string
equationexpression) throws RemoteException {
    equationexpression =
leftoperator.variable.rightoperator;
    output= equationexpression;
    mathmodel=output;

    return new GenerateMathModelResponse();
}
}
}
```

Appendix-III

Some Screen Shots of Important Interfaces of DECGrid

This screenshot shows the services running when the Globus container is started. A services called MultiDisciplinaryOptimisationService is seen running. This is the main service that performs collaboration service functions. MathService does the mathematical model building.

```

File Edit View Terminal Tabs Help
[6]: https://138.250.104.227:8443/wsrf/services/CounterService
[7]: https://138.250.104.227:8443/wsrf/services/DefaultIndexService
[8]: https://138.250.104.227:8443/wsrf/services/DefaultIndexServiceEntry
[9]: https://138.250.104.227:8443/wsrf/services/DefaultTriggerService
[10]: https://138.250.104.227:8443/wsrf/services/DefaultTriggerServiceEntry
[11]: https://138.250.104.227:8443/wsrf/services/DelegationFactoryService
[12]: https://138.250.104.227:8443/wsrf/services/DelegationService
[13]: https://138.250.104.227:8443/wsrf/services/DelegationTestService
[14]: https://138.250.104.227:8443/wsrf/services/InMemoryServiceGroup
[15]: https://138.250.104.227:8443/wsrf/services/InMemoryServiceGroupEntry
[16]: https://138.250.104.227:8443/wsrf/services/InMemoryServiceGroupFactory
[17]: https://138.250.104.227:8443/wsrf/services/IndexFactoryService
[18]: https://138.250.104.227:8443/wsrf/services/IndexService
[19]: https://138.250.104.227:8443/wsrf/services/IndexServiceEntry
[20]: https://138.250.104.227:8443/wsrf/services/ManagedExecutableJobService
[21]: https://138.250.104.227:8443/wsrf/services/ManagedJobFactoryService
[22]: https://138.250.104.227:8443/wsrf/services/ManagedMultiJobService
[23]: https://138.250.104.227:8443/wsrf/services/ManagementService
[24]: https://138.250.104.227:8443/wsrf/services/NotificationConsumerFactoryService
[25]: https://138.250.104.227:8443/wsrf/services/NotificationConsumerService
[26]: https://138.250.104.227:8443/wsrf/services/NotificationTestService
[27]: https://138.250.104.227:8443/wsrf/services/PersistenceTestSubscriptionManager
[28]: https://138.250.104.227:8443/wsrf/services/ReliableFileTransferFactoryService
[29]: https://138.250.104.227:8443/wsrf/services/ReliableFileTransferService
[30]: https://138.250.104.227:8443/wsrf/services/RendezvousFactoryService
[31]: https://138.250.104.227:8443/wsrf/services/SampleAuthzService
[32]: https://138.250.104.227:8443/wsrf/services/SecureCounterService
[33]: https://138.250.104.227:8443/wsrf/services/SecurityTestService
[34]: https://138.250.104.227:8443/wsrf/services/ShutdownService
[35]: https://138.250.104.227:8443/wsrf/services/SubscriptionManagerService
[36]: https://138.250.104.227:8443/wsrf/services/TestAuthzService
[37]: https://138.250.104.227:8443/wsrf/services/TestRPCService
[38]: https://138.250.104.227:8443/wsrf/services/TestService
[39]: https://138.250.104.227:8443/wsrf/services/TestServiceRequest
[40]: https://138.250.104.227:8443/wsrf/services/TestServiceWrongWSDL
[41]: https://138.250.104.227:8443/wsrf/services/TriggerFactoryService
[42]: https://138.250.104.227:8443/wsrf/services/TriggerService
[43]: https://138.250.104.227:8443/wsrf/services/TriggerServiceEntry
[44]: https://138.250.104.227:8443/wsrf/services/Version
[45]: https://138.250.104.227:8443/wsrf/services/WidgetNotificationService
[46]: https://138.250.104.227:8443/wsrf/services/WidgetService
[47]: https://138.250.104.227:8443/wsrf/services/examples/core/factory/MathFactoryService
[48]: https://138.250.104.227:8443/wsrf/services/examples/core/factory/MultiDisciplinaryOptimisationService
[49]: https://138.250.104.227:8443/wsrf/services/examples/core/first/MathService
[50]: https://138.250.104.227:8443/wsrf/services/examples/core/notifications/MathService
[51]: https://138.250.104.227:8443/wsrf/services/examples/core/tp/MathService
[52]: https://138.250.104.227:8443/wsrf/services/hsi/AuthenticationService
[53]: https://138.250.104.227:8443/wsrf/services/mds/test/executor/IndexService

```

The diagram below shows the services running on all the nodes of DECGrid

The screenshot shows a web browser window titled "ServiceGroup Overview - Mozilla Firefox". The address bar shows the URL: <http://isxp1315c.sims.cranfield.ac.uk:8080/webmds/webmds?infomindexinfo&xml=servicegvr>. The page content includes a "ServiceGroup Overview" section with a table of resources.

Resource Type	ID	Information
ServiceGroup	138.250.104.228	This WS-ServiceGroup has 3 direct entries, 3 including descendants.
RFT	138.250.104.228	0 active transfer resources, transferring 0 files. 0 B transferred in 0 files since start of database.
GRAM	138.250.104.228	0 queues, submitting to 0 cluster(s) of 0 host(s).
GRAM	138.250.104.228	1 queues, submitting to 0 cluster(s) of 0 host(s).
ServiceGroup	138.250.104.232	This WS-ServiceGroup has 3 direct entries, 3 including descendants.
GRAM	138.250.104.232	1 queues, submitting to 0 cluster(s) of 0 host(s).
RFT	138.250.104.232	0 active transfer resources, transferring 0 files. 0 B transferred in 0 files since start of database.
GRAM	138.250.104.232	0 queues, submitting to 0 cluster(s) of 0 host(s).
GRAM	138.250.104.227	1 queues, submitting to 0 cluster(s) of 0 host(s).
GRAM	138.250.104.227	0 queues, submitting to 0 cluster(s) of 0 host(s).
ServiceGroup	138.250.104.229	This WS-ServiceGroup has 3 direct entries, 3 including descendants.
GRAM	138.250.104.229	1 queues, submitting to 0 cluster(s) of 0 host(s).
GRAM	138.250.104.229	0 queues, submitting to 0 cluster(s) of 0 host(s).
RFT	138.250.104.229	0 active transfer resources, transferring 0 files. 0 B transferred in 0 files since start of database.
ServiceGroup	138.250.104.230	This WS-ServiceGroup has 3 direct entries, 3 including descendants.
GRAM	138.250.104.230	0 queues, submitting to 0 cluster(s) of 0 host(s).
RFT	138.250.104.230	0 active transfer resources, transferring 0 files. 0 B transferred in 0 files since start of database.
GRAM	138.250.104.230	1 queues, submitting to 0 cluster(s) of 0 host(s).
RFT	138.250.104.227	0 active transfer resources, transferring 0 files. 215.85 KB transferred in 16 files since start of database.
ServiceGroup	138.250.104.225	This WS-ServiceGroup has 3 direct entries, 3 including descendants.
GRAM	138.250.104.225	0 queues, submitting to 0 cluster(s) of 0 host(s).
new	138.250.104.225	0 active transfer resources, transferring 0 files.

The screenshot below shows submission of jobs on the condor pool.

```

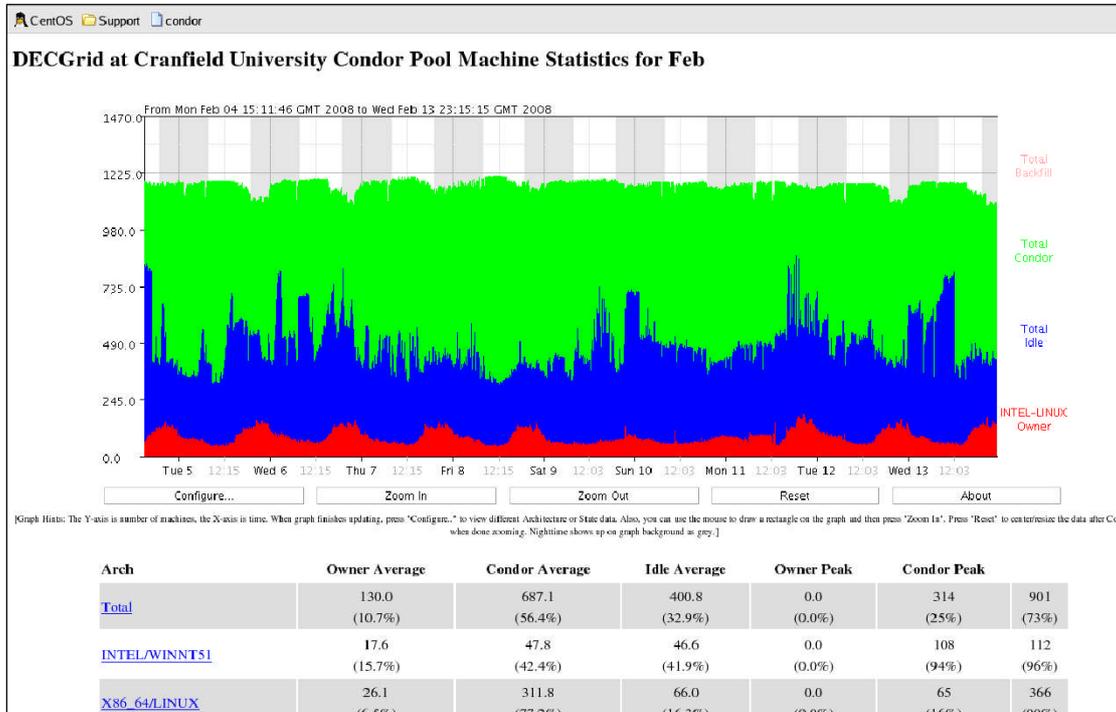
File Edit View Terminal Tabs Help
[root@isxp1313c ~]# su - condor
[condor@isxp1313c ~]$ export CONDOR_CONFIG=/usr/local/condor-7.0.0/etc/condor_co
nfig
[condor@isxp1313c ~]$ /usr/local/condor-7.0.0/sbin/condor_master
[condor@isxp1313c ~]$ export PATH=/usr/local/condor-7.0.0/bin/:$PATH
[condor@isxp1313c ~]$ condor_submit loop.cmd
Submitting job(s)....
Logging submit event(s)....
5 job(s) submitted to cluster 23.
[condor@isxp1313c ~]$ condor_submit io.cmd
Submitting job(s).
Logging submit event(s).
1 job(s) submitted to cluster 24.
[condor@isxp1313c ~]$ condor_q

-- Submitter: isxp1313c.sims.cranfield.ac.uk : <138.250.104.227:32775> : isxp1313c.sims.cranfield.ac.uk
ID OWNER SUBMITTED RUN_TIME ST PRI SIZE CMD
0 jobs; 0 idle, 0 running, 0 held
[condor@isxp1313c ~]$ condor_submit io.cmd
Submitting job(s).
Logging submit event(s).
1 job(s) submitted to cluster 25.
[condor@isxp1313c ~]$ condor_q

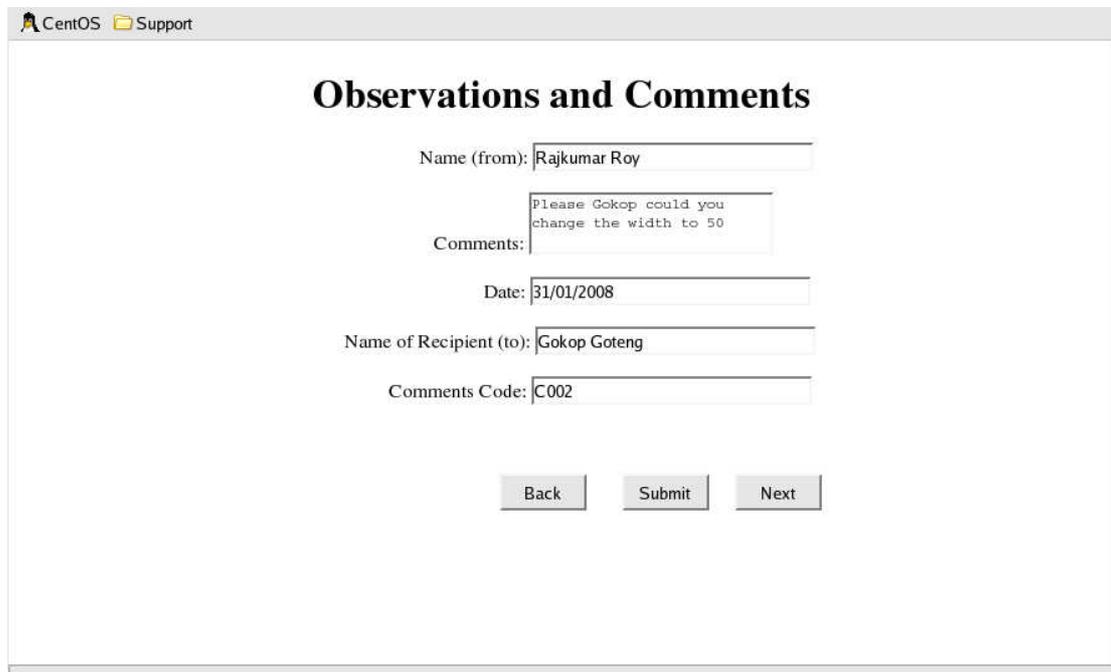
-- Submitter: isxp1313c.sims.cranfield.ac.uk : <138.250.104.227:32775> : isxp1313c.sims.cranfield.ac.uk
ID OWNER SUBMITTED RUN_TIME ST PRI SIZE CMD
25.0 condor 2/27 12:10 0+00:00:00 I 0 1.5 io.remote 200
1 jobs; 1 idle, 0 running, 0 held
[condor@isxp1313c ~]$ condor_submit loop.cmd
Submitting job(s)....
Logging submit event(s)....
5 job(s) submitted to cluster 26.
[condor@isxp1313c ~]$ condor_q

-- Submitter: isxp1313c.sims.cranfield.ac.uk : <138.250.104.227:32775> : isxp1313c.sims.cranfield.ac.uk
ID OWNER SUBMITTED RUN_TIME ST PRI SIZE CMD
25.0 condor 2/27 12:10 0+00:00:00 R 0 1.5 io.remote 200
26.0 condor 2/27 12:11 0+00:00:00 R 0 1.5 loop.remote 200
26.1 condor 2/27 12:11 0+00:00:00 I 0 1.5 loop.remote 200
26.2 condor 2/27 12:11 0+00:00:00 I 0 1.5 loop.remote 300
26.3 condor 2/27 12:11 0+00:00:00 I 0 1.5 loop.remote 300
26.4 condor 2/27 12:11 0+00:00:00 I 0 1.5 loop.remote 500
    
```

The diagram below shows the Condor monitoring tool that indicates the statistics of usage of systems in DECGrid for the month of February 2008.



The screen shot below shows the form to make observations/comments when building a mathematical model.

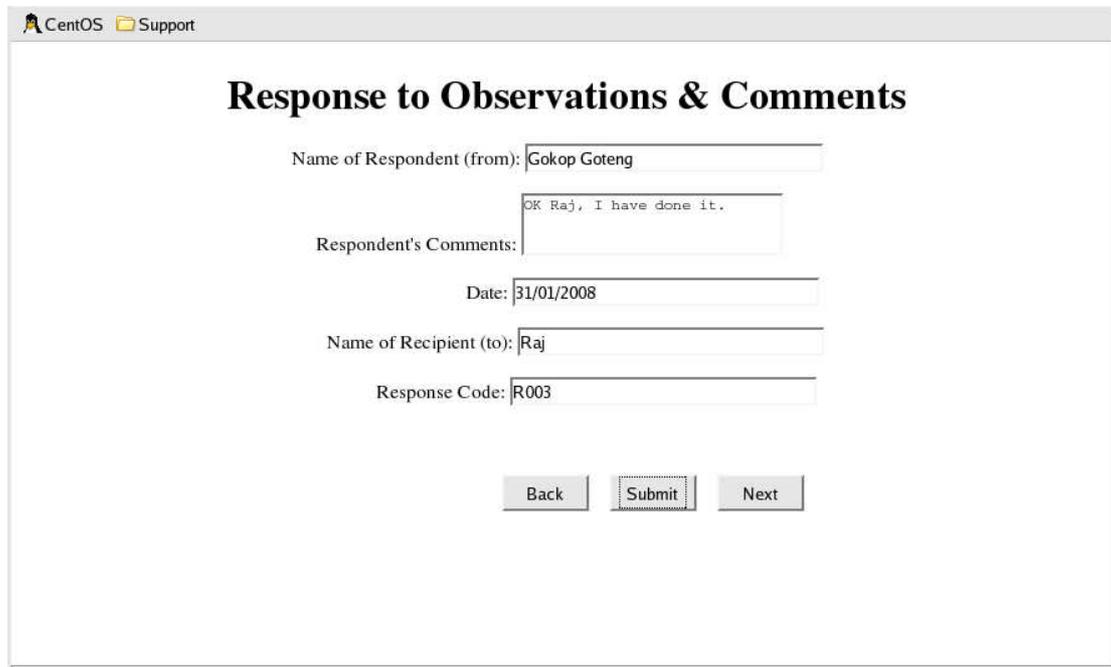


The screenshot shows a web browser window with the title bar 'CentOS Support'. The main heading is 'Observations and Comments'. The form contains the following fields and values:

- Name (from): Rajkumar Roy
- Comments: Please Gokop could you change the width to 50
- Date: 31/01/2008
- Name of Recipient (to): Gokop Goteng
- Comments Code: C002

At the bottom of the form are three buttons: 'Back', 'Submit', and 'Next'.

The screenshot below shows the form to respond to comments.

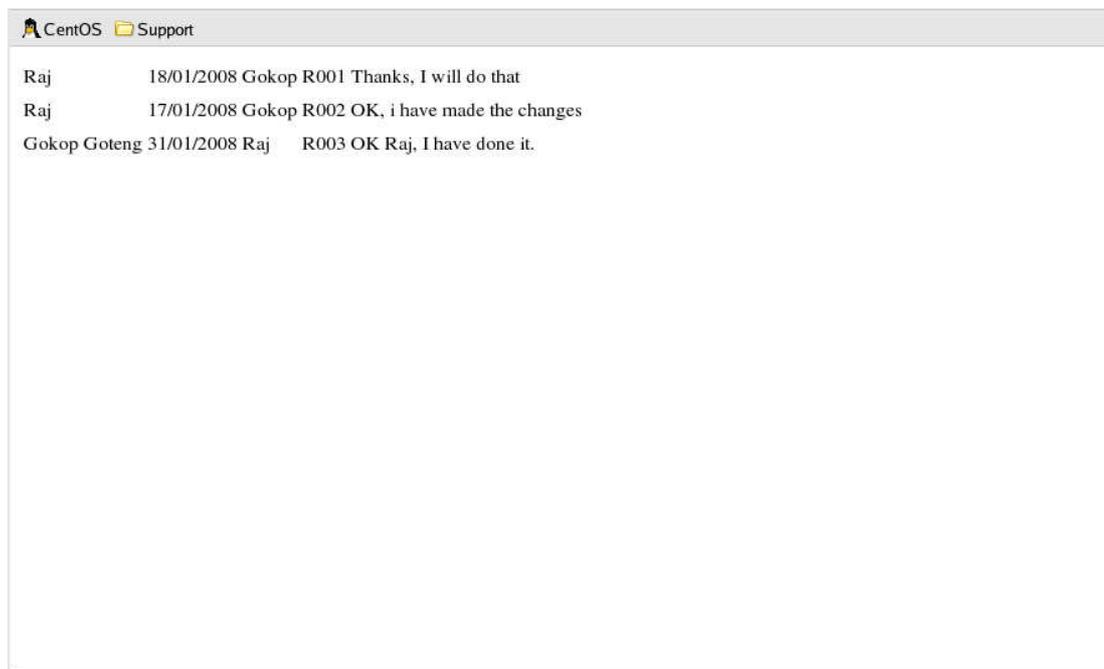


The screenshot shows a web browser window with the title bar 'CentOS Support'. The main heading is 'Response to Observations & Comments'. The form contains the following fields and values:

- Name of Respondent (from): Gokop Goteng
- Respondent's Comments: OK Raj, I have done it.
- Date: 31/01/2008
- Name of Recipient (to): Raj
- Response Code: R003

At the bottom of the form are three buttons: 'Back', 'Submit', and 'Next'.

This screenshot below is where collaborating design experts can view all their comments and responses.



Appendix-IV

DTI-Intellect Internship Preparation

As part of the internship the researcher had with the Department of Trade and Industry (DTI) Intellect and National e-Science Centre (NeSC) Edinburgh, the researcher prepared some relevant questions and ideas on how to approach the internship in line with the research.

PhD Title: Development of a Grid Service for Multi-objective Design Optimisation

Research Student: Gokop Goteng

Supervisors: Dr. Ashutosh Tiwari & Prof. Rajkumar Roy

Introduction

The first document was the introduction of the research to the DTI-Intellect Managing Director and the ICT Manager. The introduction includes summary of the research on Development of a Grid Service for Multi-objective Design optimisation (MODO). This is followed by the aim, objectives, methodology and deliverables.

Other internship commitments:

- At Intellect London: write content for the website on the outcome of the
- 1 week at NeSC Edinburg: Comparison of NeSC and my PhD
- 1 week attending the All Hands Meeting in Nottingham on 11/09/2007 to present grid solution on terrorism at Intellect's stand
- 1 week at Edinburgh Parallel Computing Centre talking about what they do and how this relates to my PhD
- Visit to National Grid Service in Edinburgh

What I want from Mark

Mark is one of the business specialists in grid computing who works closely with DTI-Intellect. The researcher used the visit to interview mark. The questions prepared are below.

1. How can I specify a grid service within SOA? I would like to have a specification document for optimisation resources that show how these resources are presented within SOA.
2. Are there different specification models within SOA? If yes, what are they?
3. I am using Globus Toolkit and Condor within Linux environment. Do you have any idea on how to register services or resources for subscription in Globus. I have tried it with little success. I am a member of the gt-user group. I get some help there sometimes.
4. What are the challenges in sharing technical data within SOA among multidisciplinary experts?
5. Is grid computing about improvement in speed and computation of processes only or are there more to this? If yes what are they and do they relate to SOA?
6. Are there better infrastructures for SOA than Globus Toolkit? What are they and the features that make them better.
7. What are the economic models used for accounting/metering charging system within SOA? I have been looking at Gridbus as one model for metering/accounting system within grid/SOA.

Summary of Meetings with Different Group Members at NeSC Edinburgh

Dr. Dave Berry, representative of GCN at NeSC took me round the NeSC building and introduced me to all staff on the first day of my visit. I immediately had meetings with Dr. Sam of the Middleware Group that same day. Sam discussed with me various middleware that NeSC uses such as gLite, NGS2, LCG and Globus. The main challenges as observed by Sam are the challenges in resolving cyclic dependencies in production grid as regards dynamic resource discovery during collaboration. I later met with other members of the Middleware Group to discuss workflow management issues within multidisciplinary optimisation. I also met with Dr. Jos and Dr. Jano. Jos described the Rapid Project he is working on with me. This is a project that aims at developing portals for easy submission of jobs. Jano on his part described to me how optimisation within multidisciplinary environment requires metadata exchange and sharing of vocabulary. By this, he means using ontology and semantic grid to describe resources and services to different experts for easy communication. I also gave a presentation on the second day titled “Grid Computing for Multidisciplinary Optimisation” to a group of about 8. Those I can remember at the presentation are Dr. Jano, Dr. Dave Berry, Prof. Jon Weissman, Miss Yin, Dr. Jos and Dr. Liang Xiu. The presentation is part of my PhD research. Comments were made on how to trim the research focus and to make it more realistic in the future.

I also had meetings with Miss Yin and Mr. Neil. Yin is a Research Assistant and PhD student working on Metadata. Yin described Metadata Resource Model Catalogue and how difficult it is to provide a generic model for all kinds of metadata. For this reason, she said it is advisable to store only references of metadata instead of the metadata. In this way, a lot of storage space is saved and a generic way of accessing the metadata is possible. Yin observed that in multidisciplinary collaboration, notification service implementation is very challenging for resource discovery to occur. Two models are used for resource discovery namely push and pull models. Yin said push model checks the life of the service while pull model sends request to host to enquire if the service is alive. Neil described in details workflow management services from service providers’ and service users’ perspectives. He maintained that service-level agreements are increasing being enforced to ensure quality of service. I also met with Prof. Jon Weissman who works on reliability issues within grid. He described the Grid Overlay Architecture which provides control nodes with redundancy information in case nodes in use depart the grid.

On the last day I met with the Training Team. This team included Dr. Guy, Dr. Mike, Dr. Boon, Mr. Ted and Mr. Luke. This team provides training for users in institutions and companies. Globus 2.5 is the main middleware used for training. Implementation issues and configuration of services were the main focus of our discussion with the training team. I also had a discussion with Dr. Liang Xiu who works on the Fire Grid Project and Nano-CMOS project at NeSC. The Fire Grid project is a demonstrator that shows the power of remote execution of resources such as simulation models and building structures. Nano-CMOS project demonstrates how databases and information can be shared in a distributed manner using GridSAM for job submissions.

Before I left, Ms Gillian Law interviewed me on my purpose of visit to NeSC and what I was able to achieve. In summary, the visit was beneficial to me in my PhD research and has further exposed me to implementation issues in grid deployment.

Implementation issues include service-level agreements, quality of service, error tolerant, workflow management of dependencies, training, scalability and metadata interchange. I now learnt that I could customise the WebMDS in Globus to provide query portal for optimisation resources. In presenting grid services, the use of Web Services Description Language (WSDL) schema is a good practice to produce a service specification document. A chapter in my PhD thesis will concentrate on comparative analysis of service specification for different problem solving environments.