Proceedings of the 19th CIRP Design Conference – Competitive Design, Cranfield University, 30-31 March 2009, pp237

Design Rework Prediction in Concurrent Design Environment: Current Trends and Future Research Directions

P. Arundachawat, R. Roy, A. Al-Ashaab, E. Shehab
Decision Engineering Centre, Cranfield University, Cranfield, UK,
{p.arundachawat, r.roy, a.al-ashaab, e.shehab}@cranfield.ac.uk

Abstract

This paper aims to present state-of-the-art and formulate future research areas on design rework in concurrent design environment. Related literatures are analysed to extract the key factors which impact design rework. Design rework occurs due to changes from upstream design activities and/or by feedbacks from downstream design activities. Design rework is considered as negative iteration; therefore, value in design activities will be increased if design rework is reduced. Set-based concurrent engineering is proposed as an alternative design approach to mitigate design rework risk, however, duplication effort for designing set of artefacts are still needed to consider before selecting set-based concurrent engineering in design activities.

Keywords:
Design Rework; Concurrent Engineering; Literature Review

1 INTRODUCTION

In concurrent design environment, there are early involvements from downstream activities such as manufacturing. The earlier involvement in concurrent engineering is known as overlapping design activities. The design lead time could be reduced by overlapping design tasks. Overlapping among design tasks could cause design reworks. Design rework inherits in overall product development lead time. Therefore, the design rework activities need to be taking into consideration for estimating the design development period.

In general, design reworks are considered as a part of iteration in every product development project. However, design reworks are considered as negative iteration [1]. Understanding the characteristics of design rework is beneficial for planning design activities. In design phase, Gantt chart or project planning network, i.e. Project evaluation review technique or critical path method (PERT/CPM), are commonly used for project planning purpose. Within project planning, design duration is assumed to be given, and normally duration in each task is inherent with rework. Value in every project is increased, if rework is removed.

The focus of this paper is to review the related literatures on design reworks. The analysis on causes of design rework and methods to estimate design rework are the outcome of this paper.

The paper is structured as follow. Design rework definition is presented in Section 2. Then, influences of preliminary design information exchanging on design rework are discussed in section 3.2. In section 4.1 to 4.3, information exchanges are discussed based on types of overlapping tasks. Tools for modelling design activities with considering overlapping and design rework prediction techniques are explored in section 4.4. Within these techniques, the impacts of design rework are concluded and discussed in table 1 and 2. Moreover at, the factors used to explain upstream changing or downstream feedback which impact design rework are summarized in table 3. Set-based concurrent engineering is introduced to reflex clearly the disadvantage of design rework. Finally, conclusion and implication for future research are presented.

2 DEFINITION OF REWORK

In construction context, rework is considered as the source deviates time planed and real progress in construction project [2]. Cooper [3] mentioned that rework is defined as error found by downstream activities. The interesting point he made is rework might be found years later after projects finished.

Love [4] concluded the definition of rework is unnecessary effort of re-doing a process or activity that was incorrectly implemented the first time. He collected the definition from previous studies most of them agree the commonality definition as quality deviation from expectation. Errors, omission, failures, damages, and change orders throughout the procurement process are concluded as caused of rework. So, rework in construction and building context is concerned on quality issue.

In concurrent design context, exploitation of preliminary information helps to reduce lead time in product development within concurrent design context. However, rework is occurred from updating of un-finalised design information [5]. This incomplete information tends to change in the later stage. Therefore, it is necessary to optimise this issue. Costa and Sobek [6] identified rework as a repeat of design under the same abstraction level. The reason for repeating is to correct error.

Rework is caused due to: (i) receiving new information from overlapped tasks after starting to work with preliminary inputs; (ii) probabilistic change of inputs when other tasks are reworked; (iii) probabilistic failure to meet the established criteria [7].

Rework is a result of proceeding tasks in parallel with using preliminary information. Yassine et al. [9] clearly developed graphical representation of preliminary information exchanged which lead to design rework in concurrent design environment as shown in figure 2. They

emphasis the downstream rework occurring due to the possibility of upstream design changes during overlapping design activities.

Another dimension of rework is a required repetition of a task because it was originally attempted with imperfect information [10].

Moreover in concurrent design, upstream rework is happened because of faults detected by downstream activity, while downstream rework occurs due to uncertainty of preliminary information given from upstream. This framework combines rework from quality deviation and information change aspects [11].

In this paper design rework is defined as unnecessary repetition of design effort. Design rework is occurred because of influences from other tasks [7], which are considered as dependency among design tasks under concurrent engineering environment. Furthermore, the design rework is uncertain or stochastic in nature [8].

Krishnan et al. [15] and Loch and Terweisch [8] are the two most cited in design rework estimation area. Lead time is assumed to be linearly converged in each rework. Therefore, changing customer requirement, which is not linearly in nature, is excluded from this area. The grahpical representations of rework in concurrent design environment are shown in figure 2b and 2c. Design rework among tasks is explained in detail in section 4.

3 PRELIMINARY INFORMATION EXCHANGE IN CONCURRENT DESIGN ENVIRONMENT

3.1 Overlapping activities

Overlapping among tasks is a major characteristic of concurrent engineering. This characteristic represents the early involvement of constituents [12]. The importance characteristic of overlapping is represented in terms of early released preliminary information. The advantage of overlapping is that it reduces product lead time and improves product innovation capabilities. Figure 1 shows total lead time reduction by concurrent development approach compared with sequential approach.

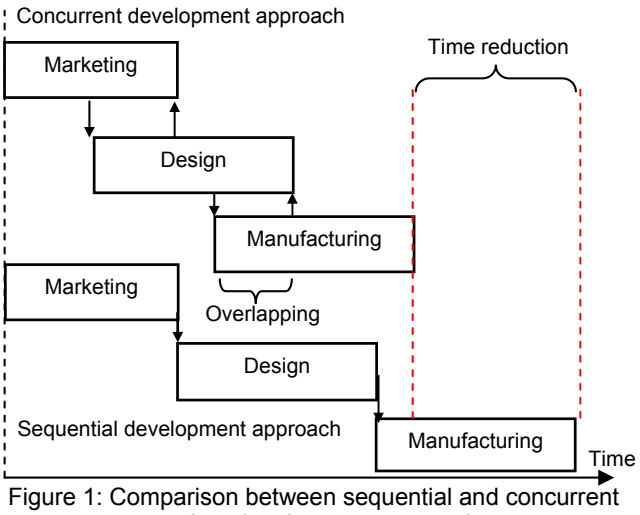

Figure 1: Comparison between sequential and concurrent product development approach

Concurrent design approach drive design activities to release preliminary information, so downstream activities could detect any faults and feedback in order to solve problems earlier. Unlike sequential design approach, each activity assumes to be complete before releasing to the next activity. For instance, if manufacturing team wait until design team completing their design, it would be more expensive and waste of time compared to faults found before hand in the design stage itself. This feedback of failure in design is caused upstream design rework.

3.2 Risks of using preliminary information

The risk of using preliminary information in overlapping design tasks is information changes. The reasons of changes are either from customer changes or from evolution of designs, etc. In Figure 1, design activity provides 'draft' design to manufacturing activity for starting development design tool earlier, however, this draft design is likely to change or update with time. Eastman [13] claimed that using advance released information bring on the issue of rework due to obsolete of data, extra time and effort to prepare to release, extra delay due to confusion, as well as bias upstream team to use conservative side of tolerances and specification. This bias impacts manufacturing difficulties and additional costs. Terwiesch et al. [14] addressed that up to 50% of total engineering capacity has to be spent to resolve rework issues. One cause of rework is from updating of preliminary information in concurrent design approach. For example, the preliminary CAD drawing could be changes after prototype testing. Therefore, the manufacturing phase needs to do some rework based on changes. The various factors that impacted design reworks are explained in section 4.

4 DESIGN REWORK ESTIMATION

4.1 Classification of overlapping tasks

The overlapping of tasks can be classified into three types as independent, dependent and interdependent [9]. The graphical model representation of overlapping tasks is shown in Figure 2.

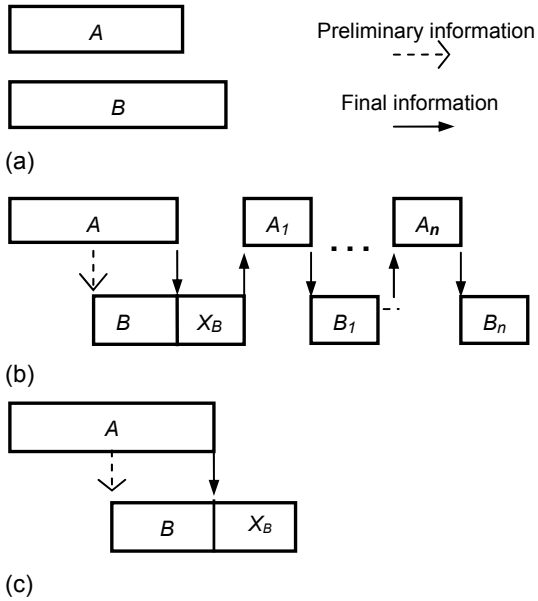

Figure 2 Pattern of tasks execution a) Independent tasks execution b) Interdependent tasks execution c) Dependent tasks execution

The advantage from independent overlapping execution is that any tasks can start freely (see Figure 2a); therefore, lead time reduction could be fully gained from concurrent approach. While, interdependent overlapping execution is defined as tasks are interaction each other, so changing from one task will cause rework to others.

Figure 2b illustrates that changing of final information from task A causes rework X_B. Task A would also be reworked, if there is feedback from task B to task A. The update of information between task A and B induces reworks,

couple of A_1-B_1…A_n-B_n, until the mutual results are satisfied. Finally, dependency overlapping execution is represented dependency of downstream design task on upstream design task only, as shown in Figure 2c. The independent overlapping execution among tasks is preferable for product design phase. The attempt to avoid interdependent and dependent design relationship is preferred, but sometime it is hard to achieve [15]. Design structure matrix (DSM) is well accepted to be a tool for dealing with complex design activities especially interdependency tasks [16]. DSM helps to re-sequence activities to avoid couple tasks by a process called partitioning, however, sometime new task arrangements is not achievable in reality [17].

Dependency among tasks is the major cause of design rework. Independent overlap among tasks is preferred in product development because any change in each activity will not impact to others.

4.2 Design rework in dependent overlapping tasks

Krishnan et al. [15] described that early exchange information to downstream could cause unnecessary iteration due to dependency of preliminary information between upstream and downstream activities. The dependency is explained by upstream information evolution and downstream sensitivities. Up stream information evolution refers to the rate of which the exchanged information reaches its final form. Downstream sensitivity is a measuring the duration of downstream work required to accommodate changes in the upstream information. Downstream rework would be huge, if a downstream activity is very sensitive to changes from upstream activities. However, this work models interaction of two activities only, while there are a lot of activities in a product development process. Downstream rework occurs because overlapping process makes downstream design task to rely on preliminary information [8]. This scenario is accounted as uncertain information. Organisation's capacity plays an importance role to reduce uncertainty during the pre-communication before starting design processes. Overlapping is risky if the upstream information may change substantially or if there is an existing of a strong dependence between activities. Therefore, changes of upstream information may cause downstream rework. The delay due to rework need to be considered trade-off between time gains from overlapping and delay from rework time. The amount of downstream rework also depends on how far downstream has already progressed. So, rework is influenced by overlapping period, pre-communication intensity. Again, this paper models rework for two overlapped activities only.

Smith and Eppinger [18] combined the benefit of design structure matrix and reward Markov Chain technique to estimate lead time. Markov chain is a method to predict the future probabilities of occurrences by analysing present known probabilities. Design structure matrix is used to identify the dependency strength among product design tasks known as repeat probabilities. However, this work is not mentioned any factors influenced to design rework. This work is very good example to considers the multi-tasks overlapping in product development project. However, the extensive discussions on the mechanism how reworks are occurred are not considered in this work.

Xiao and Si [19] combined the concept of evolution-sensitivity in information transferred [15], and the awareness of information uncertainty [8] for developing information exchange methodology. The main contribution on this paper is to prove exchanging information in batch, which they believe it could help to lower the risk from downstream rework.

Yassine et al. [9] defined downstream design rework as it is occurred by overlapping period and design changes. Downstream design task begins with knowledge accumulated by upstream design task. The knowledge accumulation is represented by probability. The design change is calculated by weighted average from probability of drastic design change and small design change. However, the probability of knowledge accumulation and probability of drastic and small design change are from expert judgments. The value in this work is the clarification of overlapped execution among design tasks, which are classified to be dependent, independent and interdependent overlapped execution. However, the model presented in this work shows the result for dependent overlapped execution only.

Luh et al. [20] defined repetition design task as uncertain task, and its occurrence can be represented by probability. However, the criteria to define of occurrence are not explained in details.

Roemer et al. [21] provided a model to calculate time and cost trade-offs of overlapping product development. The extended design time is recognized as a major risk from overlapping approach. The downstream rework is called extended design time, which is caused by the evolution and sensitivity basis. The concept of evolution and sensitivity is taken from [15]. The probability of rework occurring is non-decreasing function which is a function of overlapping period. However, the procedure to extract the probability of rework is not explained in details.

Chakravarty [22] used risk of design modified to predict downstream rework. This risk is defined by probability of incompatibility in design. The amount of rework is the result of multiplication from mapping function analogous to sensitivity, standard unit time for design or built, and risk having to modify the design work. The critical issue of this work is the factors influence the incompatibility in design, which is not considered in this work.

Roemer and Ahmadi [23] integrated the probability of rework function from [21] with the impact function. This work attempts to generate the relationships of upstream evolution and downstream sensitivity to rework. In addition, work intensity is considered as an approach to relief impacts of rework. The interactions between work intensity, and overlapping are used to calculate design lead time and cost. In this case, probability of rework is a function of upstream achievement.

Cho and Eppinger [7] assumed that task reworks are occurred from the following reasons: (1) new information is obtained from overlapped tasks after starting to work with preliminary inputs, (2) inputs change when other tasks are reworked, and (3) outputs fail to meet established criteria. The valuable part in this work is rework is classified into feedback rework and feed forward rework. Feedback rework is caused by the failure of downstream task to meet the established criteria, so upstream design task need to rework. Feed forward rework is rework that downstream task needs rework due to new information generated by upstream. This couple call iterative rework. Since, the development processes converge to its final solutions with iterative rework, there are fewer chances that new information is generated and errors are discovered. Therefore, the rework probability tends to decrease every iteration. This idea is coherent with the work from [9] in developing interdependent task modelling. However, the probability of rework and expected duration of rework are estimate by experiences.

Cascading of rework through a product development process is always issues [24]. This knowledge is used to trade off cost and schedule risk. In this work, product

development is modelled as a network of tasks, so output from one task is an input for the other task. Each rework is caused by change in particular input. However, input change are caused by either the closest upstream task itself or an impact from an upfront task, then the probability of input change is a product of multiplication from a probability of upstream changes (volatility) and a probability of a typical change causing rework from the upfront activity (sensitivity). However, all probabilities used in this model are got from experiences.

Jun et al. [25] modeled an entire product development process which includes all task patterns in reality, feedback, branch&merging, no-overlap, interaction, overlap, cycle ,and communication. The occurrence of downstream rework is estimated by non-homogenous Poisson process with fine tuning from similar historical projects, while the amount of rework estimation is based on sensitivity concept.

Overlapping of design tasks is not without risks or costs [26]. Some of the risks and costs are associated with overlapping because it initiates incomplete information for design tasks. Incomplete information in overlapping occurred from early freezing design criteria and early releasing of preliminary information and prototyping. Thought, this work suggests the approach to deal with evolution and sensitivity among design activities, but it does not provide a framework to show the amount rework that could be reduced.

Yassine et al. [5] used dynamic programming to estimate lead time with optimal information transfer policy. Too much information could extend lead time unnecessarily. Reworks are caused from time spent on outdated information, type of change (major or minor changes), and degree of sensitivity. Probability of rework is put into Monte Carlo simulation for the total rework cost incurred. However, all probabilities used in this model are got from experiences. Conclusion on factors impact design rework for dependency tasks is shown in Table 1.

4.3 Design rework in interdependent overlapping tasks

Smith and Eppinger [10] developed work transformation matrix (WTM), which is the extension of DSM, to model the design iteration process. The concept of Eigen value is used to estimate rework time. WTM can be modelled interdependent relationship among design tasks. However, the dependency data between tasks are provided by experiences engineers.

Yan and Wu [27] introduced key factors such as time, order, information, resources, and overlapping time. All these key factors are optimised by genetic algorithm (GA) before feed into the heuristic and dynamic mechanisms for scheduling purpose. Upstream design failure found by downstream is a feedback taken into account for rescheduling in their model. However, the relationships described the mechanism of rework in upstream and downstream task are not covered in this work.

Yan et al. [28] develop a branch-and-bound algorithm with heuristic rule for minimizing design lead time for concurrent product-process design activity pair. The process design's ability of discovering the faults in the product design is considered as a factor impacted rework. The authors provide the methodology to estimate mean duration, but the detail relationships between upstream and downstream are not presented in this research. Joglekar et al. [29] explored the performance of coupled development activities by proposing a performance generation model (PGM). Optimal strategies (i.e., sequential, concurrent, or overlapped) are developed with aiming to manage coupled design activities. This work is push by fixed amount of engineering resources and deadline constraints. In this work, the coupling or interdependency is modelled by performance deterioration in one task due to the rework generated by the other task. However, the practice to calculate rework in upstream and downstream is borrowed from [10].

Wang and Yan [30] modelled the iteration among an upstream product design activity and several downstream process design activities. Optimisation time and cost is a goal in this work. The distribution of faults detection by downstream is classified to be non increasing convex function and non increasing con curve function. The functions of upstream changes are assumed in the same trend. However, this model provides the optimisation framework between time and cost of product development only.

Mitchell and Nualt [32] tried to prove that cooperative planning can reduce uncertainty in concurrent design environment. Upstream rework is impacted by a lack of firm experiences in such projects, but downstream rework. Upstream changes are impacted downstream rework, which causes project delay. Rework is defined in a frequency dimension (number of change iterations) and a magnitude dimension (amount of change) relative to the original design. They study 120 business process (BP) redesign and IT development projects in the healthcare and telecommunications sectors where upstream BP design and downstream IT platform design are interdependent. The formulation of relationship is developed by using a Partial Least Squares (PLS) model and a magnitude of rework. Types of design changes are ranged from (1) incremental, (2) modular, (3) architectural, and (4) radical. However, reasons of design changing are not covered in this work.

Conclusion on factors impact design rework for dependency tasks is shown in Table 2. In conclusion, downstream design rework is impacted by upstream change, while upstream design rework is occurred due to feedback from downstream. The conclusion of factors on design rework are combined and classified into seven groups as shown in Table 3. Pre-communication is a factor to reduce upstream uncertainty [8]; furthermore; crashing or increasing intensity is a solution to compensate design rework [23], and these two factors are used to reduce the impacts of rework. Moreover, Terwiesch et al.[14] and Bogus et al.[26] proposed to use set-based design to avoid downstream design rework. Details are discussed in section 5.

4.4 Methods to estimate design rework

There are 3 approaches to acquire design rework which are direct experiment, mathematical modelling and simulation [31]. However, there is only simulation approach present in literatures. Direct experiment is hard to achieve in reality. Mathematical modelling required precise data, while product development is not [31]. Therefore, tools desired should be represented real world situation in product development in this case overlapping is compulsory characteristic needed to represented. Product development process is dynamic and stochastic, so methods to estimate design rework should allow to deal with stochastic nature. Table 1 is the conclusion of literatures related to dependent overlap execution, while Table 2 is for interdependent overlap execution. Column 2 and 3 are criteria to define factors and methods to estimate design reworks consecutively. Factors in Table 1 and 2 are concluded in Table 3. Design reworks are originated either from upstream changes or downstream feedbacks.

Authors	Factor impact rework	Criteria to define factors	Estimating methods
Hodemaker et al. [33]	Project complexity	Risk of unsuccessful integration	Stochastic model
Krishnan et al [15]	Upstream Information Downstream Iteration	Evolution Sensitivity	Non-linear program
Loch and Terwiesch [8]	Preliminary information Uncertainty Dependency	Rate of upstream changes Reduction in rate of change Impact the modifications on the downstream task	Non-linear program
	Pre-communication	Meetings before the development work starts.	
	Overlapping	Level of overlapping	
Yassine et al. [9]	Type of task dependency	Independent Dependent Interdependent	Stochastic model
	Engineering change	Major change Minor change	
	Approach used	Sequential Overlap Concurrent (fully overlap)	
Roemer et al. 21]	Incomplete information transferred in overlapping task	Probability of incorrect prediction for updated design	Stochastic algorithm
Chakravarty [22]	Information exchange Design incompatibility	Probability density function	Optimisation model
Yassine et al. [34]	Change in information	Probability of change	DSM
Browning and Eppinger [24]	Change in particular inputs	Probability direct input change Probability of the change from far upstream	DSM
Terwiesch et al. [14]	Coordinate among couple tasks	Early released information Uncertainty	Qualitative framework
Xiao and Si [18]	Information exchange Uncertainty Upstream evolution	Intensity (Non-homogenous Poisson distribution) Evolution degree	Nonlinear program
	Downstream sensitivity	Progress in downstream work	
Roemer and Ahmadi [23]	Incomplete information Crashing	Probability of rework Work intensity	Stochastic algorithm
Cho and Eppinger [7]	Update of preliminary information Impact of rework from far upstream Outputs fail to meet established criteria	Probability of rework	DSM Advance simulation
Jun et al. [25]	Information exchange Overlapping Sensitivity	Degree of overlapping Sensitivity function	Analytical model
Bogus et al. [26]	Evolution of upstream information Lack of design optimisation Insufficient design information Sensitivity	n/a	Qualitative framework
Yassine et al. [5]	Using outdate information	Time spent in using outdate information	Dynamic programming model
	Information and number of activities related	Major change Minor change	
	Sensitivity	Robustness	

Table 1 Conclusion of factors and method to estimate design rework for dependent overlap execution

From Table 1 and 2, the methods can be classified to be as qualitative and quantitative framework.

Qualitative frameworks are the works from [14] and [26], but all of them suggest the suitable scenario for point-based concurrent engineering and set-based concurrent engineering to eliminate rework impacts. Furthermore, the explanation describes on how factors impact rework are discussed rather than rework prediction. Quantitative framework can be grouped into three groups. The first one tries to classify on which criteria are impacted upstream or downstream rework [32] by using a statistical technique. The second and the third groups are related to prediction of design rework. However, the second group ([8], [15], 10], [11]) is defined as prediction tools for simplify two tasks.

Authors	Factor impact rework	Criteria to define factors	Estimating methods
Smith and Eppinger [10]	Couple of tasks Imperfect information received	Rework proportion	WTM
Yan and Wu [27]	Upstream design failure found by downstream Overlapping (Pre-release information) Feedback	Weighting factor	Heuristic scheduling Genetic Algorithm (GA)
Joglekar et al. [29]	Couple of tasks	Rework proportion	Performance generation model
Yan et al.[28]	Concurrency Risk of late detects upstream design faults. Uncertainty in input information received from upstream	Poisson process of faults discovery Magnitude of design iteration	Heuristic rules Branch and bound algorithm
Wang and Yan [30]	Downstream design discovery faults from upstream design	Probability of fault detection	Probability theory-based method (for estimating task duration) One-dimensional search algorithm
Mitchell and Nualt [32]	Lack of experiences (Impacted upstream rework)	Uncertainty	Survey research (Quantitative) Seven points Likert scale
	Cooperative planning (Impacted downstream rework)	Amount of cooperative planning	

Table 2 Conclusion of factors and method to estimate design rework for interdependent overlap execution

Factors	Sources of Rework		Detail explanations
	Downstream Feedback	Upstream Changes	
Project complexity (Integration issue)	✓	✓	Product based
Upstream design changes Evolution (Speed of continuous change)		✓	Product based Experience of designers Design support Technology
Degree of changes/updating design (Discrete changes) Major change Minor change		✓	Organisational based
Changes from far upstream (Knock on effect)		✓	Product based
Amount of overlapping task		✓	Planning based
Dependency of tasks Sensitivity		✓	Product based Process based
Faults found by downstream Chronological (early, late)	✓		Experience of designers Design support Technology
Pre-communication	✓	✓	Organisational capability
Crashing	✓	✓	Increase more resources

Table 3 Conclusion of factors impacts to design rework

Finally, the third group is a method to predict rework for multistage overlapped. However, reworks are assumed either probability distribution given or rework factors provided. So, literatures in this area are rather accounted reworks for lead time estimation than to predict it. Literatures in this area are clustered to be DSM based and non-DSM based. Works based on DSM are from [7], [24], [10] and [34], while others are [5], [21], [22], [25], [27], [28], [29], [30] and [32]. The non-DSM based tools are implemented discrete event simulation such as stochastic algorithm, dynamic programming model, branch and bound algorithm, genetic algorithm, heuristic algorithm, etc. (details in Table 1 and 2), to predict lead time of product development.

Table 3 is the conclusion of factors initiated rework from table 1 and 2. They are considered from the point of of a particular design activity. For example, if designers are designing a subsystem, their solution could be impacted

either from the predetermined subsystems (upstream) or the subsequence subsystems (downstream). So, coordination and communication among team menber are critical in product design and development. If the members are not coordinate to solve the integration issues and communicate to update each one result, it could increase unnescessarily rework in the project.

5. REDUCING RISK OF REWORK BY SET-BASED CONCURRENT ENGINEERING

Changing in upstream phases in product development processes can alter not only upstream but also downstream activities. Sources of changing can be either from higher level strategic decision (market changed) or operational decision (constrains in technical aspect) [35]. Variation of design solutions due to all these source of changing is account as uncertainty in product development [8]. Ward et al [36] revealed how TOYOTA's automotive development team deal with uncertainty in car development. They also point out that lead time of development in TOYOTA is less than the other US automotive manufacturers significantly. The key difference is what TOYOTA implements set of possible design solutions and then narrow the alternatives down in parallel until achieve satisfied design, while the US use only one solution. If there are mistakes, they have to either solving with a lot of rework or getting new solution to work with. The US practice is relied on one single "point", so it can be mentioned as point-based concurrent engineering. Ward et al. [36] conclude that the set-based paradigm is the key importance to deal with uncertainty and ambiguity occurred in automotive development.

Terwiesch et al. [14] argued that set-based concurrent engineering is suitable not only for uncertainty but also ambiguity situation. However; design planners need to consider the duplicate cost and information starvation cost against rework cost. Therefore, this issue needs to be proved mathematically.

Bogus et al. [26] proposed to use set-based concurrent engineering to reduce sensitivity of downstream design activity. However, finding from [36] contrast this propose, because TOYOTA development teams tend to use set based for all levels in car development projects by using a lot of prototypes.

Ford and Sobek [37] compared set-based concurrent engineering and point-based concurrent engineering development time and cost by using real option concept. Product development is divided into three phases, conceptual design, system design and detail design. There are 2 probability types needed for calculation, probability of generate change initially and probability of discover change need. Each design phase is addressed with probability number, after that put them into real option calculation. The convergence time results are calibrated with the survey data from industries. However, time convergence can be model, but the detail characteristics of set-based concurrent engineering are not mathematically modelled in this work. There are various aspects needs to be considered before implementing set-based concurrent engineering, e.g. performance of team members Engineers in TOYOTA work for more than one vehicle development projects [36]. Other aspects are involvement of suppliers, controlling of chief engineers, organisation, etc. Therefore, the comparison of cost and time between set-based concurrent engineering and point-based concurrent engineering needs to be considered the whole context of product development process.

Set-based concurrent engineering is proposed to solve a downstream rework issue [26], which is directly eliminated dependency of downstream from upstream activity (details in factor 4 Table 3). Set-based concurrent engineering is claimed as one key success of Japanese compared to US auto industries. However, to change from 'point' to set paradigm need to be investigate more.

6. CONCLUSION IMPLICATION FUTURE RESEARCH

Most of literatures implemented simulation based approaches to illustrate design rework embed in design lead time. The major finding is most of which prefer to put "probability" of changes or feedback into the models to represent rework, while the factors impacted rework are qualitatively explained rather than expressed mathematically in the model. Furthermore, the contexts of implementing concurrent engineering such as, performance of team members, involvement of suppliers, controlling of chief engineers, organisation, tools used, etc., need to be considered.

Concurrent engineering approach could be implemented with higher efficiency in design activities, if reworks are lowered. Sources of design rework in concurrent engineering are from upstream design changes and downstream feedbacks. Furthermore, rework is particularly occurred when one single design choice is selected (point-based concurrent engineering). It is necessary to understand and estimate design rework in point-based concurrent engineering and estimate duplicate cost and starvation cost in set-based concurrent engineering for select product development approach.

Based on the understanding of design rework from upstream changes and downstream feedbacks, all factors addressed in Table 1 to 3 will be used for design rework estimation. This estimation will be based on analogy approach. This approach allows putting multi factors in estimation framework by using the techniques call pair-wise comparison [38]. The one outcome from rework estimation is helping development team to select between 'point' or 'set' of designs in product development, which is lack in [18]

7. REFERENCES

[1] Ballard, G. 2000, Positive VS Negative Iteration in Design, Proceedings of the International Group for Lean Construction 8th Annual Conference (IGLC-8), Brighton, UK

[2] Friedrich, D. R., Asce, M., Daly, J.P. and Dick, W.G., Revision, Repairs and Rework on Large Projects, Journal of Construction Engineering, 113, 3: 488-500

[3] Cooper K. G. 1993, The Rework Cycle: Benchmarks for the Project Manager, Project Management Journal, 25, 1: 17-21

[4] Love, P. E. D. 2002, Auditing the Indirect Consequences of Rework in construction: a case Based Approach, Managerial Auditing Journal, 7, 3: 138-146

[5] Yassine, A. A., Sreenivas, R. S. and Zhu, J. 2008, Managing the Exchange of Information in Product Development, European Journal of Operational Research: 311-326

[6] Costa, R., and Sobek II, D. K. 2003, Iteration in Engineering Design: Inherent and Unavoidable or Product of Choices Made?, Proceedings of DETC'03 ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference

[7] Cho, S. H. and Eppinger, S. D. 2001, Product Development Process Modelling Using Advanced Simulation, ASME 2001 Design Engineering Technical Conferences and Computers andInformation in Engineering Conference Pittsburgh, Pennsylvania September 9-12: 1-10

[8] Loch, C. H. and Terwiesch, C. 1998, Communication and Uncertainty in Concurrent Engineering, Journal of Management Science, 44, 8: 1032-1048

[9] Yassine, A. A., Chelst, K. R. and Falkenburg, D. R. 1999, A Decision Analytic Framework for Evaluating Concurrent Engineering, IEEE Transactions on Engineering Management, 46, 2:144-157

[10] Smith, R. P. and Eppinger, S. D. 1997, Identifying Controlling Features of Engineering Design Iteration, Journal of Management Science, 43, 3: 276-293

[11 Mitchell, V. L. and Nault, B. R. 2007, Cooperation Planning, Uncertainty, and Managerial Control in Concurrent Design, Journal of Management Science, 53, 3: 375–389

[12] Koufteros, X., Vonderembse, M. and Doll, W. 2001, Concurrent Engineering and Its Consequences, Journal of Operations Management, 19, 1: 97-115

[13] Eastman, R. M. 1980. Engineering Information Release Prior to Final Design Freeze, IEEE Transaction on Engineering Management, EM-27, 2: 37-42

[14] Terwiesch, C., H. Loch, C. H. and Meyer, A. D. 2002, Exchanging Preliminary Information in Concurrent Engineering: Alternative Coordination Strategies, Journal of Organization Science, 13, 4: 402-421

[15] Krishnan, V., Eppinger, S. D. and Whitney, D. E. 1997, A Model-Based Framework to Overlap Product Development Activities, Journal of Management Science, 43, 4: 437-451

[16] Browning, T. R. 2001, Applying the Design Structure Matrix to system Decomposition and Integration Problems: A Review and New Directions, IEEE Transactions on Engineering Management, 48, 3: 292-306

[17] Browning, T. R. 1998, Using of Dependency Structure Matrices for Product Development Cycle Time Reduction, The 5th ISPE International Conference on Concurrent Engineering: Research and Applications, Tokyo, Japan, July 15-17: 1-8

[18] Smith, R. P. and Eppinger, S. D. 1997, A Predictive of Sequential Iteration in Engineering Design, Journal of Management Science, 43, 8: 1104-1120

[19] Xiao, R. and Si, S. 2003, Research on the Process Model of Product Development with Uncertainty Based on Activity Overlapping, Journal of Integrated Manufacturing Systems: 567-574

[20] Luh, P. B., Liu, F. and Moser, B. 1999, Scheduling of design projects with uncertain number of iterations, European Journal of Operational Research, 13: 575-592

[21] Roemer, T., R. Ahmadi and Wang, R. 2000, Time-Cost Tradeoffs in Overlapped Product Development, Journal of Operation Research, 48, 6: 860–865

[22] Chakravarty, A. 2001, Overlapping Design and Build Cycles in Product Development, European Journal of Operation Research, 134: 392–424

[23] Roemer, T., A. and Ahmadi, R. 2004, Concurrent Crashing and Overlapping in Product Development, Journal of Operations Research, 52, 4: 606–622

[24] Browning, T. and Eppinger, S. D. 2002, Modelling Impacts of Process Architecture on Cost and Schedule Risk in Product Development, IEEE Transaction on Engineering Management, 49, 4: 428–441

[25] Jun, H. B., Ahn, H. S. and Suh, H. W. (2005), On Identifying and Estimating the Cycle Time of Product Development Process, IEEE Transaction on Engineering Management, 52, 3: 336-349

[26] Bogus, S. M., Molenaar, K. R. and Deikmann, J. E. (2006), Strategies for Overlapping Dependent Design Activities, Journal of Construction Management and Economics, 24: 829-837

[27] Yan, J. H. and Wu, C. 2001, Scheduling Approach for Concurrent Product Development Processes, Journal of Computers in Industry, 46: 139-147

[28] Yan, H. S., Wang, Z. and Jiang, M. 2002, A Quantitative Approach to the Process Modelling and Planning in Concurrent Engineering, Journal of Concurrent Engineering: Research and Application, 10, 2: 97-111

[29] Joglekar, N. R., Yassine, A. A., Eppinger, S. D. and Whitney, D. E. 2001, Performance of Coupled Product Development Activities with a Deadline, Journal of Management Science, 47, 12: 1605–1620

[30] Wang, Z. and Yan, H. S. 2005, Optimising the Concurrency for a Group of Design Activities, IEEE Transactions on Engineering Management, 52, 1: 102-118

[31] Smith, R. P. and Morrow, J. A. 1999, Product Development Process Modelling, Journal of Design Studies, 20: 237-261

[32] Mitchell, V. L. and Nault, B. R. 2007, Cooperation Planning, Uncertainty, and Managerial Control in Concurrent Design, Journal of Management Science, 53, 3: 375–389

[33] Hoedemaker, G. M., Blackburn, J. D., and Wassenhove, L. N. V. 1999, Limits to Concurrency, Journal of Decision Sciences, 30, 1: 1-17

[34] Yassine, A. A., Whitney, D. E. and Zambito, T. 2001, Assessment of Rework Probabilities for Simulating Product Development Processes Using the Design Structure Matrix, ASME 2001 International Design Engineering Technical Conferences Computers and Information in Engineering Conference, Pittsburgh, Pennsylvania

[35] Sobek II, D. K., Ward, A. C. and Liker, J. K. 1999, TOYOTA's Principles of Set-Based Concurrent Engineering, *Sloan Management Review,* Massachusetts Institute of Technology

[36] Ward, A., Liker, J. K., Cristiano, J. J. and Sobek II, D. K. 1995, The Second TOYOTA Paradox: How Delaying Decisions Can Make Better Cars Faster, Sloan Management Review, Massachusetts Institute of Technology

[37] Ford, D. N. and Sobek II, D. K. 2005, Adapting Real Options to New Product Development by Modelling the Second TOYOTA Paradox, IEEE Transactions on Engineering Management, Vol. 52, No. 2, p. 175-185

[38] Saaty, T. L., 1994, Fundamentals of Decision Making and Priority Theory with the Analytic Hierarchy Process: 6, RWS Publications, Pittsburgh, PA.