Proceedings of the 19th CIRP Design Conference – Competitive Design, Cranfield University, 30-31 March 2009, pp51

Artificial Neural Networks to Optimize the Conceptual Design of Adaptable Product Development

J. Feldhusen, A. Nagarajah
Chair and Institute for Engineering Design (ikt), RWTH Aachen University, Steinbachstrasse 54B, Aachen, 52074, Germany
feldhusen@ikt.rwth-aachen.de, nagarajah@ikt.rwth-aachen.de

Abstract
This paper describes an approach to apply Artificial Neural Network (ANN) to support the developer to optimize the conceptual design of Adaptable Product Development. The ANN learn from the previously mapping between the requirements and the product solutions. For a new development Process of an Adaptable Product the ANN analyse existing solution and sort them according to their probability of success.

The advantages of this approach is on the one hand that ANN is able to identify the best suitable product solution for a new development order and on other hand the ANN automatically store the knowledge of the developer.

Keywords:
Product Development Process; Engineering Design; Neural Network; Adaptable Products

1 INTRODUCTION

In today's automotive supplier industry majority of the products are developed by adaptation or by varying already existing products. This kind of Product Development is named as Adaptable Product

Development. One reason for such a Development can surely be found in the demands of the supplier to reduce cost and time required for Product Development on an ongoing basis [1]. Thereby, the enterprises face the dilemma of depending on existing solutions on one hand but on the other hand to develop innovative solutions to enhance their own competitive ability compared to the international competitors. To solve this conflict, the developers have to be supported to automate the Processes, which are very time consuming but not sophisticated in respect of creativity. One of this process is searching for already gathered, but insufficiently documented knowledge [2].

What is today's mode of operation of the product developers for an adaptable product development? From numerous research projects in the automotive industry it can be derived that the developer does not select the suitable parent product which will fulfil the new customer requirements but selects the last product version as a base for the new development. Therefore the development is aggravated unnecessarily by the missing knowledge. An expert system can produce relief. The inauguration of these systems is very critically discussed in the industry due to the fact that those systems give the impression to be able to substitute the human developer in doing his tasks and therefore making him dispensable. This can be seen as one reason for the less-proliferation of those systems in the industry [3]. Thus such an expert system, which has to be developed, has not only to be technically successful, but also has to cope with the social environment of an enterprise. To solve this problem a NN based expert system seems to be success-oriented. This system will be teached with the knowledge of the developers by mapping the parent product solutions with associated requirements. In figure 1 the approach is described. For the handling of the requirements in a NN based system it is essential to collect them completely and to quantify them, i.e. to describe them with an unequivocal numerical value for representing in computer. Hence the requirement description should occur with the help of a visualisation language, which should lead to a better communication with the customer. To workout such a language it is necessary to subdivide the requirements into elementary constituents. After the requirements have been determined and described for a certain product, these should be related to the realised product. This mapping should be done for all products developed in the past. The NN based system then shall be teached with these mappings.

For a new development order, which is given in principle by changed values of the requirements (in fig. 1 by regulator denoted) the existing solution with the most potential to solve the new order can be identified.

2 STATE OF THE ART

In first stages of the product development process a preferably complete, fast and consistent evaluation of the requirements is essential for the success of a product. But in this stage a part of the requirements has limited use in the form in which they are defined in the conceptual formulation. It is incumbent upon the developer to prepare these requirements purposeful [4].

In today's automotive supplier industry, it can be observed that the developer orients himself on already existing solutions when developing a new product. He uses methods, e.g. benchmarking or reverse

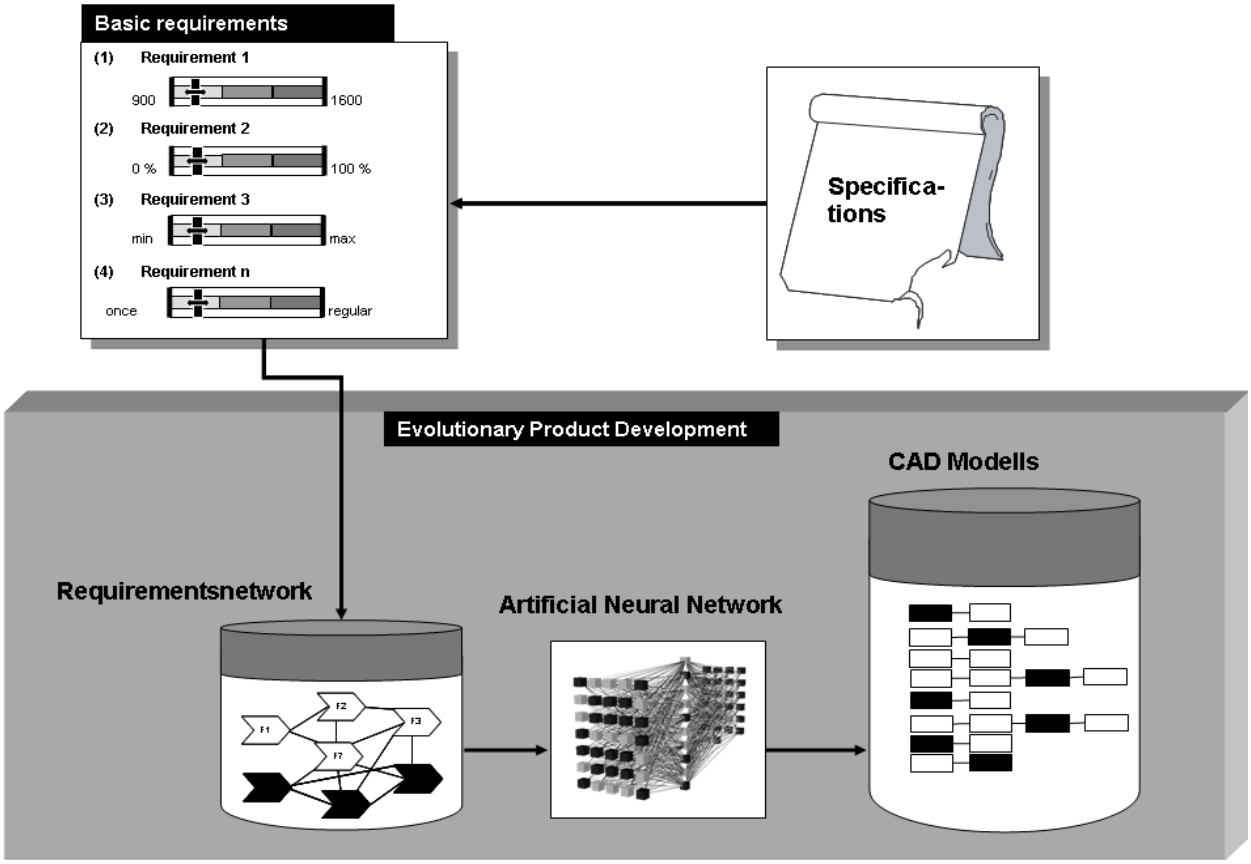

Figure 1: The adaptable Product Development

engineering, to get the necessary knowledge from his own products or from competitor products. Using these methods, it is required that the developer has both identified the functions fulfilling the customer's requirements and that he is able to deduce the functions from the design. In the design all information about its development is stored. By analysing the shape of a design, the restrictions can be identified that could arise in later stages of a Product Development. In spite of using these methods for identifying the requirements, it can be over and over detected that requirements haven't been correctly clarified or have to be amended afterwards [4]. That this is still a large problem for the automotive industry can be seen through numerous publications regarding this topic [5].

In the IT industry, software developers use the visual-specification language UML (Unified Modelling Language) for communication with customers. The demand for the use of UML in other industrial sectors led the Object Management Group (OMG) to introduce the modelling language SysML for commonly used technical systems. This modelling language has to be examined for the possibility of migration for communication with customers in the field of Product Development in the automotive industry. Moreover, the modelling language requires a basic description of the requirements in the type of attributes. For instance the specifications that are submitted in the written text form in the practice have to be transferred to a basic type of description. In this context the VDA (german automotive industry association) guideline can be used as a basis. It describes the requirements by condition, subject, demand word, object and action [6].

Artificial Neural Networks recreate natural Neural Networks like the brain. Neural Networks are used e.g. for pattern recognition and creation of language, as Artificial intelligence for games and the optimization of Processes. In the field of engineering science, they are utilized for the forecast of costs using modular construction systems or for the calculation of constructive parameters. They are consulted if forecasts are deduced from a large number of historical data empirically [7]. An application of the Neural Networks in Product Development, and especially for linking the requirements with product components, doesn't exist.

3 THE APPROACH OF THE RESEARCH PROJECT

For a successful procedure of this research, it is necessary to divide the main goal into the following three sub goals:

1. The first aim is do describing requirements in the type of basic requirement modules. Based on these modules a graphical „language" for visualization of requirements should be developed.

2. Development of an Artificial Neural Network that supports the developer in a meaningful way with the accomplishment of the Adaptable Product Development for new requirements, by offering purposeful parent CAD models (representing parent product solution), which should be suitable for the intended requirements.

3.1 Procedure for the creation of basic requirement components

A basic prerequisite for the aimed goal is the creation of a description of the requirements, which should be as unambiguous as possible. The first step according to the VDA should be a type of description for the requirements in form of basic components (Fig. 2).

The basic components, which should consist out of subject, object and predicate should be set-up especially for the type of business and should be stored in a database. The user should only use these defined components for the description. It should be ensured that errors, which may arise during the communication with the customer, would be minimized. A specification sheet with a description of the requirements in text form comprises a high error potential due to its large space for interpretations. This would be enhanced by an elementary form of description, but can't be seen as the optimal solution due to the fact that like this, the behaviour of the system can only be identified very hard.

A graphical visualization of the requirements would lead to a further optimization of the communication with the customer.

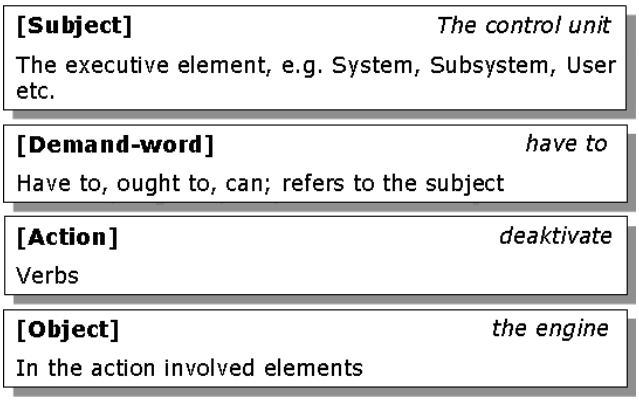

Figure 2: Basic requirements components

Complicated facts of the case can't be described generally understandable in form of plain text. In that case graphical visualization helps a lot. Also graphics contain a defined syntax with defined semantics. The composition of graphics of that kind results in a model of reality, which can easily be transformed to others. This model has to be understandable for all people involved in the development Process.

The largest accumulation of models that are able to describe commonly used systems was developed within the Object Management Group (OMG): We talk about the SysML (Systems Modelling Language). The functional limitations of the systems are described in SysML by the means of Use-Case-Diagrams. Use cases have been developed, in order to register the functional limitations of the systems, i.e. the functional requirements. At this point we have a complete picture: Every functional requirement of the specification sheet can be assigned to one use case. Derived from the other requirements, the use case can be linked to protagonists (which could be persons or systems), who will interact with the system and the pre- and post conditions that are results of the functional requirements.

The requirement components are categorized for the compilation of a graphical shape in a first step. The relationships between the classes should be described with attributes like „is affiliated with"and „in contradiction to "(Fig. 3).

When developing a new type of product, it can be checked if the ascertained customer requirements can be assigned to an already existing class of requirements. Therefore possible contradictions should be identified at the beginning of the development Process. If a class of requirements is non-existing until now, a „dummy"-class should be generated. The „non-existence"of a requirement is a first hint that there is no technical solution within the own company to realize that requirement.

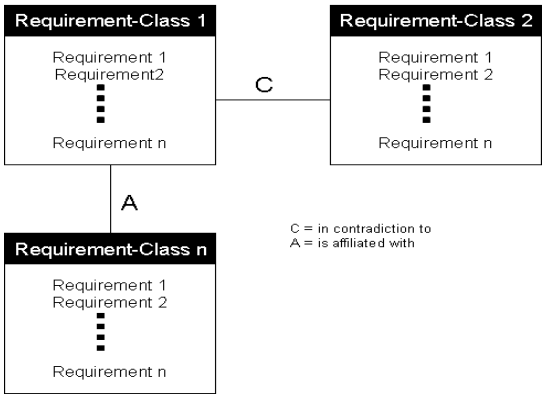

Figure 3: Relationship between requirement classes

On the basis of these classes of requirements, a graphical modelling language for this type of industry can be developed, following the graphical modelling language SysML. As mentioned above, it is possible to define the desired feature of a product within a model of requirement classes and with a use case model; the desired behaviour can be visualized (Fig. 4).

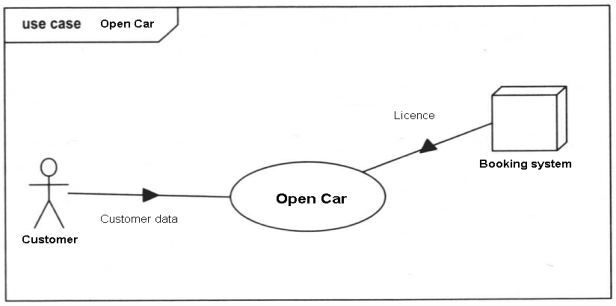

Figure 4: SysML - Use Case for open a car

Every type of business has its own language and symbols. These should be used for the modelling language. For an effective communication, the system has to be seen like an organism. After a first visualization of the requirements by means of the modelling language, the illustration has to be developed to get a more detailed view using further interactions with the customers.

Within the graphical presentation the possibility exists to update changes of the requirements during the Process of a Product Development, using the already existing illustration. Moreover, the illustration should be assembled hierarchically using a top-down approach.

3.2 Development of an Artificial Neural Network

Artificial Neural Networks are Networks consisting of artificial neurons. They are one arm of Artificial intelligence and in principle object of research of the neuroinformatics. The origin of the Artificial Neural Networks can be found – as well as Artificial Neurals –

in biology. They are compared with the natural Neural Networks, which are forming cross-linking of nerve cells in the brain and the spinal cord. Altogether it is about an abstraction (construction of a model) of information Processing and less about the replicating of biological Neural Networks [8]. Artificial Neural Networks mostly base on the cross-linking of many McCulloch-Pitts-neurons or slight modifications of them. In principle, other Artificial neurons can be used in Artificial Neural Networks, e.g. the High-Order-neuron. Depending on its task, the Network topology (the assignment of connections to knots) has to be reviewed very well. After construction of a Network a training phase follows. In this period the Network will „learn". Theoretically, a Network can learn using the following methods [8]:

- Development of new connections, deleting existing ones
- Change of the weights (the weights wij of neuron i to neuron j)
- Adaptation of thresholds of neurons
- Adding or deleting neurons

Moreover, the learning behaviour changes when changing the activating function or the neurons or the rate of learning of the Network. Practically, a Network mainly „learns" by modifying of the weight of the neurons [8]. An adaptation of thresholds can be done by a Bias-neuron (bias means to distort in this context). That's why ANNs are able to learn complex non-linear functions by a „learning"-algorithm that tries to identify all parameters (influencing factors) of the function out of existing input- and desired output values by iterative or recursive course of action. Thus ANNs are a realization of implicit learning (so called connecting paradigm), because the function consists of many elementary similar parts. First as a total the behaviour will be complicated.

Perceptron

The perceptron (named after the English word percept) is a simplified Artificial Neural Network, which was firstly presented by Frank Rosenblatt in 1958. In its basic version (simplified perceptron) it consists out of one single artificial neuron with adaptable weights and a threshold. Among this concept several combinations of the original model are perceived, differentiated between one-layer and multi-layer perceptrons. The basic principle is to convert an input vector to an output vector, therefore forming a simple associative memory.

One-layer perceptron

A one-layer perceptron has only one single layer of artificial neurons that represent the output vector at the same time. Therefore, every neuron is represented by a neuron function and receives the entire input vector as a parameter. The Processing occurs completely similar to the so-called Hebb's rule for natural neurons. However, the activating factor of this rule is substituted by a difference between set point and actual value. Due to the fact that the Hebb's (learning) rule relates to the weight of the individual input values, the learning of a perceptron is done by adapting the weight of every neuron. If the weights have been learned once, a perceptron is able to classify input vectors that are slightly deviating from the vector learned. Just therein, the perceptron's desired ability to classify exists, from which it owes its name.

Frank Rosenblatt showed that a simple perceptron with two input values and one singular output neuron, can be used for the representation of the simple logic operators AND, OR and NOT. Marvin Minsky and Seymour Papert demonstrated in 1969 that a one-layer perceptron is not able to dismantle the XOR-Operator (linear separation problem).This lead to a termination of the research on the Artificial Neural Networks (ANN).

Multi-layer perceptron

Later on, the limitation of the one-layer perceptron was solved by the multi-layer perceptron, which besides the output layer, also has at least one further layer of hidden neurons (so called „hidden layer"). All neurons of a layer are completely linked to the neurons of the next layer forward (so called feed forward-Networks). Further topologies have also proved themselves:

- Full-Connection: The neurons of one layer are connected with the neurons of all following layers.
- Short-Cuts: Some neurons are not only connected with all neurons of the next layer, but additionally with more neurons of the next but one layer.

Amongst others, a multilayer perceptron can be trained with the back propagation algorithm. In this algorithm, the weights of the connections are changed such that the Network can classify the desired patterns after a supervised learning.

Method of learning

Back propagation or Back propagation of Error [1] is a widespread practice for learning Artificial neuron Networks. Although first verbalised in 1974 by Paul Werbos, it first became well-known by the paper of David E. Rumelhart, Geoffrey E. Hinton and Ronald J. Williams from 1986 and led to a „renaissance" of the research on Artificial neuron Networks.

It belongs to the group of supervised learning methods and is exerted as generalization of the „Delta" rule on multi-layer Networks. In addition, it is necessary to have an external teacher at every moment of input knowing the desired output, called target value,. The back propagation is a special case of the common gradient method in optimization, based on the median quadratic error.

With the „learning problem"for arbitrary Networks a transformation of given input vectors to given output vectors as exact as possible is aimed for. For this purpose, the quality of the transformation is described by an error function.

The aim is now the minimization of the error function, whereas in general only a local minimum can be found. The learning Process of an Artificial neuron Network takes place with the Back propagation-method by adaptation of the weights, due to the fact that the output value of the Network is – beside the activating function – directly dependent on them.

The algorithm for Back propagation can be divided into the following phases:

- An input pattern is applied and propagated forward through the Network.

- The output of the Network is compared with the desired output. The difference between these two values is considered as the error of the Network.
- The error is now propagated backwards, from the output layer to the input layer where depending on their influence on the error, the weights of the neuron connections, will be changed. This guarantees a convergence to the desired target output when attaching an input again.

3.3 ANN for Adaptable Product Development

In Chapter 3.2 the essential basics to design an NN, which is able to fulfil the requirements of this approach is described. The neural network will be building up with a set of multi-layer perceptrons and should be trained by the developer.

The intention of this approach is to design a system, which should optimize the development process of an Adaptable Product. Therefore the system to be developed should be trained by the procedure of the developer. It is important to recognize which requirement has led to which product characteristic. For a new product development the product characteristic can be derived immediately, if is it possible to recognize these assignments. But these assignments are so complex that they can not be identified without supporting by adequate tools. The developing system should be trained by the developer, because he determines the assignments.

Due to Artificial Neural Networks the Adaptable Product Development should be designed more effective and efficient. Artificial Neural Networks comply a continuous learning of thinkable and desired product components according to a customer's requirement. Therefore the requirements should be prepared for the implementation in a NN system. The quantifiable requirements like cross-section, forces, etc. can be processed relative easily. The qualitative requirements like ergonomics, haptics etc. are difficult to process. In the first step these requirements should be transferred in numerical values by estimating of experts. The requirements, which belong to a product, are represented as a vector. Hence the different values by different order lead to different requirement vectors.

Due to this approach it is possible to detect a proposal for the product component that fulfils essential customer requirements, but is not glutted by unnecessary versions.

By implementing a Neural Network, the required development time for familiar requests should be shortened effectively. By this, it is possible to automatically check if the requirement can be realized by the existing variant of products, as soon as the requirements are determined, thus if the solution of the requirement is within the solution range (Fig. 5).

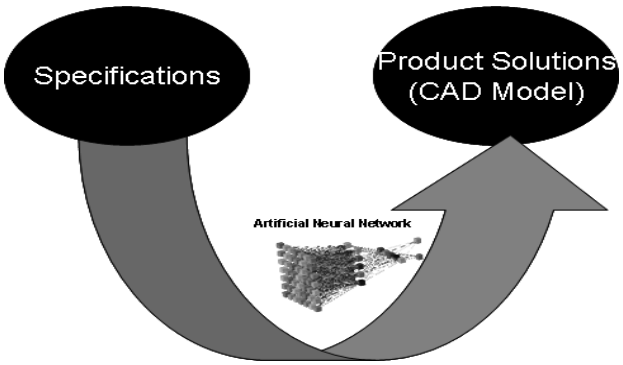

Figure 5: Mapping of specifications and product solutions

The results would be visualized to the user - sorted by their probability of success – in form of an intuitively operable system for proposals.

4 SUMMARY

The goal of this research project is to apply a NN based system to short the time for conceptual design of an Adaptable Product Development. The shortening of the conceptual design phase should be achieved by applying the NN-based system to identify the existing solutions that are suitable for a new product order. The identified solutions for the new product requirements should be adapted with a minimal effort on research and design. Applying the NN System for support the conceptual design phase of product development is the novelty of this approach. An important precondition for this application is the quantification of the requirements. To apply the requirements to a NN based system they should be quantified. The quantifiable requirements can be processed relative easily. The qualitative requirements are difficult to process. In the first step these requirements should be transferred in numerical values by estimating of experts. In a further step, the fuzzy modelling should be used to these requirements to get numeric values automatically.

A further precondition for the success of this approach is surely the complete identification of requirements. This is to develop a visualisation language for a better communication with the customers. In an Adaptable Product development there are not various differences between product generations. Therefore the requirements can be assumed directly in the visualisation language. For this the modelling language SysML which stems from informatics should be adapted to suit the needs of the automotive industry. The basic for the design of such a language is the description of the requirements in elementary components. For this the VDA (german automotive industry association) guideline can be used as a basis.

5 ACKNOWLEDGEMENTS

The authors like to thank the German Federal Ministry of Education and Research for aiding this research project.

6 REFERENCES

[1] Ehrlenspiel, K., Kiewert, A., Lindemann, U., 2007, Cost-Efficient Design, Amer Society of Mechanica

[2] Feldhusen, J., Nurchaya, E., Loewer, M., 2007, Variant Creation Using Configuration Of A

Reference Variant, ICED International Conference On Engineering Design, Paris, France, 28-31 August

[3] Rammert, W., Schlese, M., Wagner, G., 1998, Wissensmaschinen: soziale Konstruktion eines technischen Mediums: das Beispiel Expertensysteme, Campus Verlag

[4] Kruse, P., 1996, Anforderungen in der Systementwicklung, VDI-Verlag

[5] Almefelt, L., Andersson, F., Nilsson, P., Malmqvist, J., 2003 Exploring Requirements Management in the Automotive Industry., ICED International Conference On Engineering Design, Stockholm

[6] Verband der Automobilindustrie, 2006, Automotive VDA-Standardvorlage Komponentenlastenheft, Frankfurt a.M.

[7] Pulm, U, 2004, Eine systemtheoretische Betrachtung der Produktentwicklung, PhD thesis, Munich

[8] Zell, A., 2000, Simulation neuronaler Netze, Oldenburg Verlag