Creating an ability to respond to changing requirements by systematic modeling of design assets and processes

S. André¹, F. Elgh¹

¹Department of Product Development, Jönköping University, Jönköping, Sweden
(samuel.andre@ju.se, fredrik.elgh@ju.se)

Abstract - System suppliers, e.g. original equipment suppliers, are important for the success of many products. They design a unique solution, often in close collaboration with other companies, based on different product concepts and/or core technologies. The solution can then be manufactured in different quantities depending on the client's need. High level of customization is required as the interfaces are not standardized, the performance is not negotiable, requirements are not initially fixed and the specific system interacts with, is affected by, or affects other systems that are simultaneously developed. A system supplier commonly designs and manufactures solutions for different OEMs and must support many models and variants in their product portfolios. Efficiency, short lead-time, continuous technology development, and adaptability are essential for the competitive edge. A product platform approach has been a success for many companies to enable variety at low cost, however, it is not applicable for system suppliers. This work describes the result from a case study where a platform approach enabling a new way of structuring, publishing and managing design assets and processes was introduced at a company with the purpose to improve the ability to respond to changing requirements in the quotation process and the subsequent product development activities.

Keywords - Keywords: customization, system suppliers, platform, product development, changing requirements, DSM

I. INTRODUCTION

The original equipment supplier (OES) industry differs from other sectors in that the customer is often already involved in the quotation stage. This is in stark contrast to companies developing off-the-shelf products for the end consumer market and where the level of customization is low. It is not uncommon for products to be developed in cooperation with the customer, which is often an original equipment manufacturer (OEM) or another supplier which is part of a larger supply chain, and for projects to run for several years. During the cooperative development phase, requirements are often added, removed or changed. This has been investigated in the automotive industry by Almefelt, Berglund, Nilsson, and Malmqvist [1] and is said to be a natural process since knowledge is gained and prerequisites change throughout the project. Andersson [2] states that misunderstandings are one source of requirement changes, often due to the fact the requirements are not specified clearly at the outset.

A. Background and associated literature

Customization is here referred to as abilities and strategies that aim at the design and manufacture of tailored products for individual customers. Depending on where customization actually starts, four different business models have been proposed: engineer-to-order, modify-to-order, configure-to-order, and select variant [3]. For the latter two, product platforms have gained substantial success as enablers for efficient customization. Platforms have been widely accepted in the industry to serve a wide variety of products while maintaining business efficiency. Early descriptions of product platforms focused on efficiently providing the market with a product variety while also keeping internal variation low and in that way reach a higher level of standardization of the production [4]. Platforms have also served as a way to reach different customer segments efficiently and simultaneously by featuring commonality in product components and interfaces. Recent research has focused on platforms using a more abstract definition; these platforms aim to reuse more of the skills and knowledge (i.e. assets) created in a given company in order to reach higher efficiency during development. Although only a small amount of research has been conducted on companies developing highly customized products at the supplier level, a platform formulation for a supplier using small batch production has been investigated by Berglund, Bergsjö, Högman, and Khadke [5]. These authors conclude that a modular platform is not feasible in such a company and that most of the reuse is found at a higher level of abstraction. From that perspective, Johannesson [6] has asked whether companies can afford not to adopt a platform approach. Levandowski, Jiao, and Johannesson [7] propose a two-stage model of an adaptive product platform by using a combination of modular and scalable architectures to manage changing requirements during early phases. Suppliers are also the focus in [8] where knowledge modeling is applied using design structure matrices (DSM) in the development of design automation systems to support efficiency in the quotation process.

However, how OES can benefit from a platform strategy in the face of the realities described above has not yet been fully investigated, and methods to support this are few in the literature. Platform approaches tend to require a focused development of the platform and late customer involvement, which in turn requires some knowledge about which future variants are to be derived from the
platform. This is simply not possible for OES since interfaces are not standardized, the performance is not negotiable, requirements are not initially fixed and the specific system interacts with other systems that are simultaneously developed by other suppliers and the OEM itself.

B. The need of an expanded platform approach for EOS

Previous research has introduced [9] and applied [10] the design platform (DP) concept in order to make it possible for OES to harvest the benefits of platform strategies. In previous research, focusing on platforms [11] and design repositories [12], an emphasis has been put on the artifact, leaving out the design knowledge required for the process which leads up to the artifact. The industrial objective of this research is to support EOS to work platform-based and thus gain efficiency when designing highly customized products. The scientific objective is to add to knowledge by proposing a general platform approach for companies where previous research has left a gap [5] which also has been explored by others [7]. The contribution of this paper consists of adding an expanded relationship view, including the process, to the platform approach applied in [10] and thus support efficient customization in the quotation process and subsequent activities at the same time as being responsive to changing requirements.

C. Methodology

The Design research methodology (DRM), proposed by Blessing and Chakrabarti [13], have been applied throughout this work. The methodology describes four stages: Research clarification (RC), Descriptive study I (DSI), Prescriptive study (PS), Descriptive study II (DSII). This article is positioned in the PS of the research project since it involves the development of a methodology and the application of a tool which is based on an industrial and scientific gap which has been identified in RC and DSI. The research project has involved 4 companies developing customized products which all have been subjected to in-depth interviews, document reviews, method development and presentation of applications. The companies have participated in developing success criteria which have been used to guide and evaluate the research project outcome in a qualitative manner. The method and tool described in this paper have been introduced, tested and reviewed in one of the involved companies.

II. A PLATFORM APPROACH FOR OES DEVELOPING HIGHLY CUSTOMIZED SYSTEMS

The platform approach allows for different levels of abstraction to be used in the platform definition in order to coherently tie a heterogeneous mix of company assets to the platform. Instead of letting the platform only describe how and what components and sub-systems are to be assembled, as with most product platform concepts available [11], this platform approach model aims to also support in the designing of these components and sub-systems. It is described in detail in [10] and will be shortly summarized in the following.

The platform approach builds on an object-oriented view of the product portfolio. The UML representation of the platform approach can be seen in Fig. 1. The generic product item (GPI) is the class which describes a certain generic product. The GPI has a top part attribute which can be either an Assembly or Component, depending on the type of product which a company develops. The Assemblies and/or Components which populate the GPI can be seen as empty placeholders which are to be filled with design knowledge in different formats. The building blocks which have been proposed to be associated with the Assemblies and Components are Design Elements (DE) which have been inspired by the ICARE forms described by Stokes [14]. DEs are discretized blocks of knowledge descriptions and correspond to four specific classes:

- **Entity** is a description of an existing embodied component or subsystem; it includes attributes such as function, behavior, and links to geometry models.
- **Activity** is used to describe a task or process that often includes an execution order; attributes include inputs, outputs, triggers, and objectives.
- **Rule** describes a guideline or a set of valid relations for the designer to employ; rules can be described by mathematical formulas, tables, or in text form, and rules can describe design parameters and how they affect different variables.
- **Constraints** can be one of two types; internal constraints describe limitations that are usually based on some boundary condition, such as manufacturing equipment, while examples of external constraints are customer requirements and legal requirements.

In addition, an association class is used to specify the
relationship type in between the different DEs. The Relation class can either be a Process relation or Resource relation, both having the attribute of rational specifying why the specific relation exists. The Relation class will be further outlined subsequently.

The platform approach covers the expand and use stage. The first refers to when the model is setup or modified. The use phase refers to when the information in the model is used by instantiation of the generic product items and browsing of the platform content.

B. The extended relationship view

To support how the content of the GPIs and instances are to be used together, an extended relationship view has been developed. Multiple relations exist between DEs, Assemblies, and Components which are hard to model using only the product structure tree. The relations which become obvious in the tree structure are hierarchical and focuses on how systems are grouped together. Within a certain group of a hierarchical level, additional guidance is needed e.g. to support in what sequence to perform Activities or which knowledge that can be used as a resource in order to perform a specific Activity. The relationship view is composed of inter-domain Design Structure Matrices (DSM) for modeling relationships between objects of the same type, and multi-domain matrices (MDM) where objects are of different types. An explanatory extended relationship matrix is shown in Fig. 2. The relationships that are connecting the different objects are objects themselves (Fig. 1.) and instantiated from the following types:

- **Process relationship** states sequence order of DEs. It can refer to the order in with a set of activities should be executed or the order in which components are to be designed.
- **Resource relationship** states if a certain DE is to be used as a resource for another DE. E.g. a DE describing a parametric CAD model can be a resource for an Activity DE describing the design of a specific component.
- **Hierarchy relationship** are derived from the tree structure and are not an explicit relationship type in the class diagram (Fig. 1.) since it is implicitly existing in the aggregation relationships between Part and DesignElement as well as the recursive aggregation on the Activity class.

Constraints can usually not be related to each other within the domain since assumptions regarding the solution need to be made which do not exist early in a development project. For later phases, when solutions are known, the hierarchy of constraints can be broken down into Assemblies and Components. Due to that, Constraints do not have a domain of their own but are solely inputs and thus part of MDMs. Similarly, no DSM exist of the Entity domain since they are considered as the output of the process with their hierarchal relationships described by Assembly and Component DSMs and MDMs. The remaining domains are explained in the following:

- **Activity domain (AD)**
  The AD consists of all the Activities which have been formalized and associated to a GPI or an instance. Within the domain, Process relationships are used to specify the specific execution order of the activities.
- **Rule domain (RD)**
  The RD specifies if there exist specific relationships between the rules. Most likely is that a rule must be used as a resource for another rule.
- **Assembly domain (AssyD)**
  The AssyD states the hierarchical relationships between Assemblies. These are the same as can be viewed in a traditional tree structure.
- **MDMs**
  The MDMs are used for mapping relationships in between the different objects of different domains. The possibilities become many, but in general, there are three parts: inputs, outputs and the process itself. The AD is the core of the relationship view and either require an input, output or both. The Constraints and Rules primarily act as inputs since they state what constraints need solving and what rules to use in the process. The AssyD and MDMs associated with Components state the hierarchy or tree structure as well as relations to DEs that can be used for their realization. Entities are the output of the process specified by the set of Activities and are therefore primarily related to those. Entities are also related to Assemblies and Components since they embody the solution.

III. CASE OF APPLICATION

In order to support and evaluate the expansion and use of the platform approach the Design Platform Manager (DPM) support tool was developed. DPM was introduced at an industrial company in order to support the application of the platform approach. The company is acting as a second-tier automotive supplier and employs 600 people at the studied site and 10,000 people worldwide. The focus was on a business area in which the company
develops and manufactures highly customized car interior subsystems. One product concept developed within this business area is a pneumatic seat support system, the main aim of which is to vary the pressure distribution between the body and the seat so as to increase comfort and ergonomics. Inflatable air cells are used to create the varied pressure distribution. The other main part types of the system consist of the carrier plate, valves, a pump, and tubes.

DPM was used to model an array of GPIs and variants. Several DEs was also identified and formalized using spreadsheets and associated with the GPIs and variants by use of DPM. This is described in detail in [10].

The user interface of the software tool is shown in Fig. 3. DPM enables the user to model GPIs as trees whose main structures are composed of Assemblies and Components as holders of DEs (1, 2). Different GPIs can be viewed, added, and altered (3). The variants belonging to a specific GPI are shown in (4); these are the variants that have been instantiated using a specific GPI as a base structure. The properties of objects in the user interface can be viewed and altered in (5). The content of the PDM system can be searched through an integrated PDM viewer (6) and associated with the modeled structure (1). Further, GPIs, Assemblies, Components, and variants are all saved, retrieved, and updated using specific DPM tables in the PDM database.

In addition to the above-described functionality, an extended relationship view has been added in DPM. An instance is shown in Fig. 4, where the Activity and Assembly domains are visible. Examples of relationships can also be seen whereas the activity domain is also visualized as a network in Fig. 5. The extended relationship view allows for visualizing a GPI or an instance with associated domains and MDMs. Relations are modeled by specifying relation objects in the cells. The empty DSMs, MDMs, and hierarchical relations are automatically created from the information existing in the GPI. Additional relationships are modeled manually. As seen in Fig. 5, two parallel process flows are seen. Since hierarchical relations have been filtered out for visual purposes, the process flows might not be independent as it may seem. The figure, in this case, is only for visualizing an example. The formalization of the design process might also not be complete, which is common in the case company, which leaves gaps in the process.

An additional functionality which have been integrated is sensitivity analysis through analyzing change propagation in the matrices. By “poking” a row in the matrix the related objects are identified in a recursive manner. This ultimately produces a network to the user of affected DEs.

IV. DISCUSSION

The modeling of relations between DEs and structural levels is associated with the platform expansion stage. This takes place during the development and in the reflective stages by the addition of DEs and relationships as new knowledge emerges. The use phase is relevant in early stages, e.g. quotation, or when changes to requirements occur throughout the development. In the quotation phases, the DPM is used to search for similar GPIs and related instances which corresponds to the requirements of the customer. If similar designs are found there is also a process modeled, in the extended relationship view, for each GPI which in these cases can be applied. To the process, there are also resources attached in the form of rules and constraints which can be used to design new solutions where there are no ones to reuse. This ultimately supports the process of designing customized products by
the reuse of design knowledge on different levels of abstraction.

Changes in requirements are common in this industry. When changes occur, the sensitivity analysis enables the identification of what solutions that are affected when requirements change and what activities that have been performed resulting in the solutions. This ultimately can propose possible countermeasures and activities to be performed in order to respond to the changes. It also builds an awareness of the complexity that changes have which in turn offers the possibility to proactively reduce the number of relations between clusters of solutions.

The tool DPM has been applied to company-specific data and verified throughout the research project by the author. It has been presented and partly evaluated by company representatives through workshops, discussions and by using questionnaires. The methodology and tool receive good marks in terms of functionality, adding to the possibilities of reusing knowledge and supporting the designer. The user interface needs improvements to make the use of the tool user-friendly and simple.

V. CONCLUSIONS AND FUTURE WORK

This paper has presented a platform approach with associated generic product structure- and extended relationship view and modeling possibilities. The aim is to support OES to apply a platform approach by the introduction of extended relationships between the objects which populate the platform. This enables the modeling of process- and resource relationships which are important for OES developing customized products where the final products cannot be known beforehand. The tool has been partly presented to the company and receives good marks.

The network view provides possibilities for the future to use clustering and other graph- and layouting techniques. When several instances have been created there also exist possibilities to add logic in order for the DPM application to propose suitable courses of action depending on the input of the user.

DPM and associated method will be applied and tested in more companies to further strengthen generalization, usefulness, and applicability.

ACKNOWLEDGEMENTS

This article was partly supported by the Swedish Agency for Innovation Systems, VINNOVA within the ChaSE project. The authors would like to express their sincere gratitude to the case company for sharing invaluable knowledge.

REFERENCES

3. Hansen, B.L., Development of industrial variant specification systems. 2003, Technical University of Denmark: Danmarks Tekniske Universitet, Department of Management EngineeringInstitut for Planlægning, Innovation og Ledelse.