7th Workshop On Product Structuring – Product Platform Development
Chalmers University Of Technology, Göteborg, March 24-25, 2004

DYNAMICS OF PRODUCT PLATFORM LIFECYCLE AND DISPOSITION MANAGEMENT

Timo Lehtonen, Tero Juuti, Antti Pulkkinen, Asko Riitahuhta

Abstract

According the present thinking, a function based platform approach is most virtuous for configurable products. Although there are successful cases of utilising this type of platform in industry, success has often been only temporary. This is due that challenges set by long-term platform life cycle management has not been addressed adequately. Dynamic Modularisation (Dymo) is a business and product development paradigm, which aims to add the processes that are needed in handling the life-cycle variation to the platform-paradigm. However achieving the dynamic platform and getting rid of the static nature normally associated to modular structures is much more easily proposed than implemented. The cause-effect chains in product structure evolution are numerous and dispositional changes are not effecting only from design phase to later product life-cycle phases, but changes in for example subcontracting could require re-planning of the product structure. In this paper cause-effect chains in product structure evolution are examined. The attention is focused to requirement management and release planning processes, which can be seen as key processes enabling dynamic platform management.

Keywords: Design management, Configurable products, Platform lifecycle, Modularisation, Product families

1 Introduction

Customer Variation required in many business areas causes explosion of product variants and sets challenge to product structuring. Modularisation is often seen a solution for this. However a modular structure is not a solution in itself, because there exist different types of modular structures and different approaches to deploy them [2]. According the present thinking, a function based platform approach is most virtuous for configurable products. Although there are successful cases of utilising this type of platform in industry, success has often been only temporary. This is due to that challenges set by long-term platform life cycle management has not been addressed adequately.

Dynamic Modularisation (Dymo) is a business and product development paradigm, which aims to add the platform-paradigm the company processes that are needed in handling the life-cycle variation [1,2]. In Dymo the product development work is made on two levels. On the first level aim is to develop and maintain platform, which addresses to business needs. The most of product development work is made in module development projects that are not directed to one particular product, but the aim is to strengthen or widen the platform capabilities. Dymo includes the following areas of interest:

- customer requirement management

- product architecture management
- product architecture development
- module creation process

In the module creation process, the suitable modules are developed for fulfilling customer requirements. All these actions are targeted for creating a product platform, which enables launching a product family, which corresponds to market needs now and in predictable future. Product platform is a common set of re-usable assets used in developing a set of products that from a certain point of view constitute a product family.

The second level in Dymo-process is integration of final product releases. The product programme selects suitable configuration of modules to meet customer expectations as well as maintaining profitable business case. Product programme integrates only ready-made modules into product release. For this reason product programmes activity is not developing products, but rather integrating them.

Practical implementation of Dymo-process is shown in figure 1. Starting from the left we see how product architecture is formed according the business needs. The available platform has capabilities that are constrained by system level architecture. Subsystems are derived from the system level architecture. Modules/Components are released for the product program according to the subsystem architecture. Product program integrates ready-made modules into product release. For this reason product programs are not called "development" but "integration".

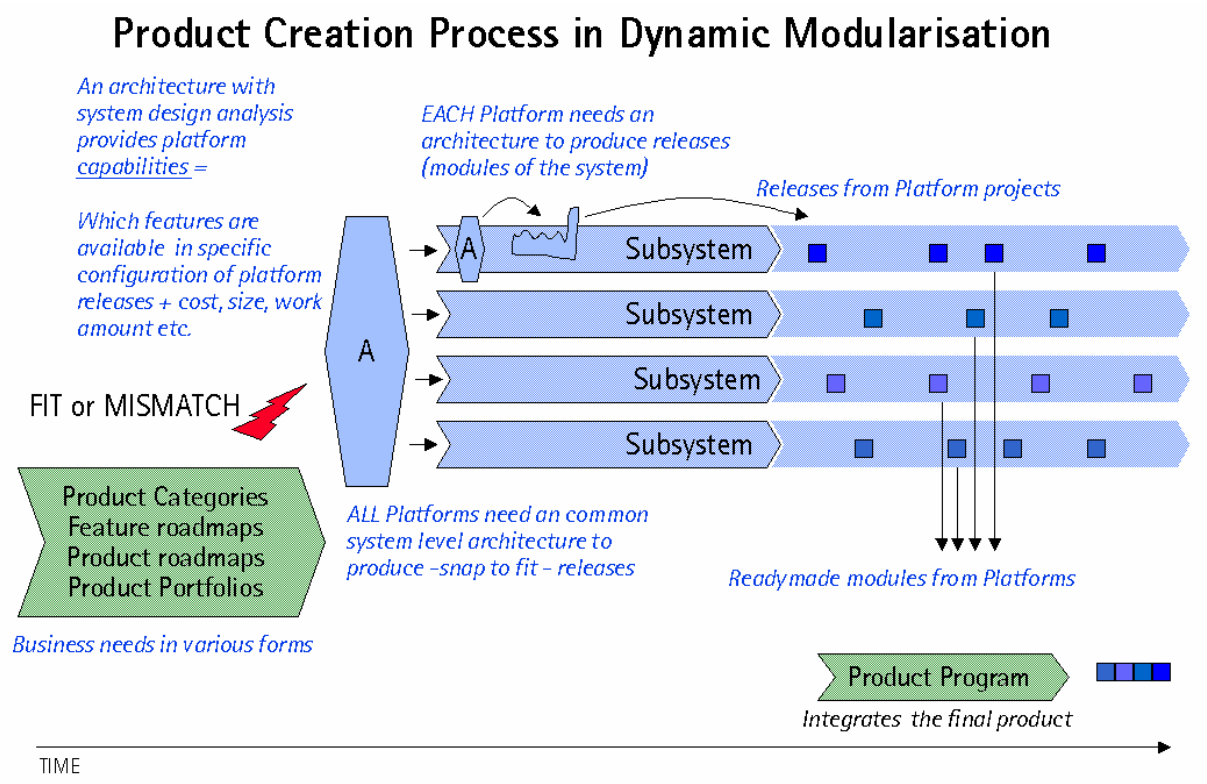

Figure 1: The implementation of Dymo-process in practice.

The major difference and challenge to platform based product development is the dynamic module structure. Currently the perception is that the platform module structure has to be static to be manageable and beneficial for the company. The Dymo addresses the operative needs in managing variety of drivers causing changes in module structure.

2 Achieving the dynamic platform

Getting rid of the static nature normally associated to modular structures is much more easily proposed than implemented. Usually, a platform is formed according the current situation in company product offering. This way of working might well create a product family, which supports actions made in yesterday and is adequate still today. However, a platform made this way faces eventually the need of renovation or even replacement. By keeping on thinking of static platforms, we will end up to the conclusion that platforms are most suitable way of working in areas where the rate of change is slow. However, a real paradox is that in these kinds of slow motion business areas the need and the possible gains from the platforms are minimal compared to areas where the rate of change is high and therefore need for design re-use more urgent. Therefore, it seems that platform is a tool, which is most easily applicable when it is hardly needed, but difficult to use there where it would really give business benefits.

The challenges set by changes can be identified and solution can be found, if we examine the cause-effect chains in product structure evolution. In figure 1 there is an overview picture where these cause-effect chains are drawn by causal/cognitive mapping method [3]. The overview picture shows the complexity of the matter. The net of relations should be considered as a meta-model and all cause-effect relations shown are not equally relevant for all modules. The mechanisms of dynamism can be traced. Then cause-effect chains are analysed in taking account the life-cycle phases of the platform. The focus should be targeted to dispositional mechanisms. Normally dispositions are thought to be directed from design phase to later life-phases of product. The DfX-methods and concurrent engineering are seen as a solution to cope with these effects [4]. However there is dispositional effects working also other way round. For example dividing the production to subcontractors sets requirements for design of the product and could for example require of using assembly based modules instead of function based modules.

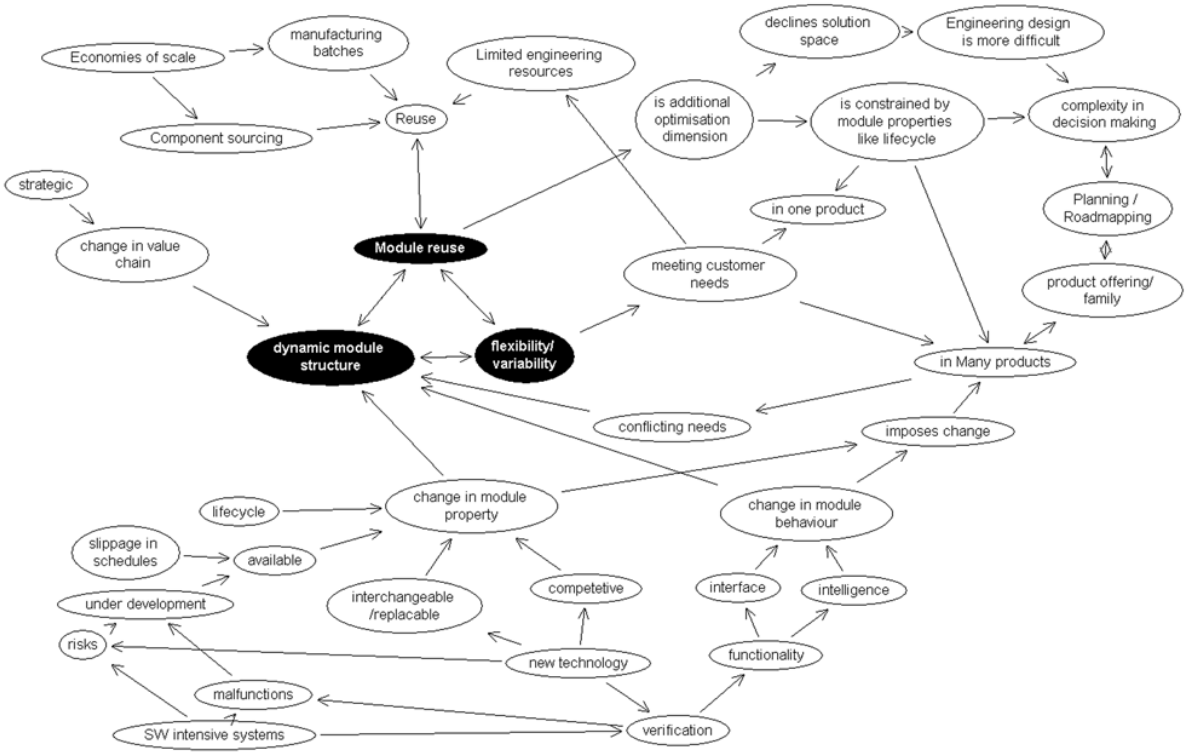

Figure 2: The cause-effects chains in product structure evolution. Although arrows are directed, long-term life cycle management requires ability to be able estimate dispositional effects also in back-wards direction.

At the first glance, the cognitive map of product structure evolution may seem obscure. However there are only six relations to "*dynamic module structure*", which in this case is the essence of product platform. So we'll have to concentrate on these relations.

First relation is with "*change in value chain*". The practical example of this is when company changes to such supplier, which is able to deliver highly integrated module instead of 3 separate modules from two different suppliers. This saves effort in supply management and in integration, for example.

The second relation covers "*module reuse*". The reuse of design can be seen the main factor in achieving cost savings. This relation is very straightforward, "the more – the better".

Next relation "*flexibility*" is connected to platform capabilities. Is platform able to meet the customer needs? Also next relation "*conflicting needs*" is addressed to platform capabilities, but the viewpoint is different. Can the platform meet all requirements simultaneously? The flexibility needs to be defined against the business environment. For example if the differentiation point gives competitive edge to the company the flexibility needs to be assessed against such criteria.

Changing module properties belong to the area of engineering problem solving. It is notable that a customer is actually not directly related to this and the changes come from other sources like availability and strategic-issues.

Changes in module behaviour have an effect to some interfaces of modules. In this area are for example re-allocations in functionality and control.

In teaching design of modular products, the functionality is accepted as a usual starting point. However, when we look figure 2, we found that most things that affect the dynamic structure

are related to business environment. Therefore, the knowledge of own business environment is crucial when a company is about to build dynamic platforms. This knowledge in itself is not applicable, but it is basis to forming two key processes to guide platform evolution. These processes are

- market requirement management
- new product property release planning (road mapping)

3 Requirement management

Market requirement management is by no means a new issue in product development. However, when we are aiming flexible satisfaction to requirements, we have to take a different attitude. Instead of surveillance and data collection, the process should be more active. The possible knowledge of your platform capabilities must be available in this process and all new requirements must be considered according to this background. If company is major player in the market, the attitude should be even pro-active – in other words, the company should actively try to affect the tastes and wishes of market. Seeking and observing of the market trends might in yesterday been a task for marketing only. Now all gathered information must be evaluated at once and it must be translated it the form "what this means to us".

Keeping the product offerings tight in within own platform capabilities might sound too restrictive strategy. However, there are success stories in history to be found. Scania was in dubious situation in 70's with buss chassis production [5]. One of their important products was bus with a rear engine. Originally, their product was designed for urban transport and it had traverse engine in the back. This layout concept was not very virtuous in tourist and intercity buses. Scania realised that the engine should be longitudinal with reference to the chassis and they launched two products for this segment. Unfortunately, Scania lacked suitable engine. So, they launched two products. The cheaper BR85S had too underpowered engine to be considered as an excellent product. The more clamorous BR145 had 14-liter V8 engine, with plenty of power but with very high price tag and high fuel consumption. The smaller BR85S was suitable "prices starting" –model and the BR145 was suitable prestigious top model, but the actual seller between them was missing! And what did Scania? They kept on selling these two products and that's all. Taking account the inadequacies of these models, they sold very well. And it must be remembered that in business of Scania, the buses are minor factor and the trucks are the majors. If a bus model can't be assembled from truck components, then it is not made. One could think that this policy would become fateful for bus department within long term. On the contrary, Scania has been very successful bus maker and still today is. In this case keeping the profitability of the business has been more important than fulfilling exactly the requirements of market. In addition, this is the lesson, which should be learned within platform paradigm also.

4 Release planning

The other process needed in "steering" dynamic platform is product property release planning. When market requirement management process decides what properties there will be in our products, the task of release planning is to decide when these properties are available. The document, which contains the release dates, is often called roadmap and that is why this process is called road mapping. In Dymo, way of working the actual product

development work is made in module projects. In this case, roadmap shows releasing the new modules. The release can be made in multiple phases. First, the intention of releasing a module is declared and preliminary specification is given. Next phase is releasing the information about the module interface with integration notice. Then the final properties and actual release day can be given. All product integration projects, whose production commences after that release day, can now select this module to be used in their product.

One important aspect is to synchronise and re-plan the releases in such way that customer gets what is promised regardless the changes in actual technical content of the releases. In other words the idea is to make sure fully functional products meeting customer needs can be made out of platforms releases.

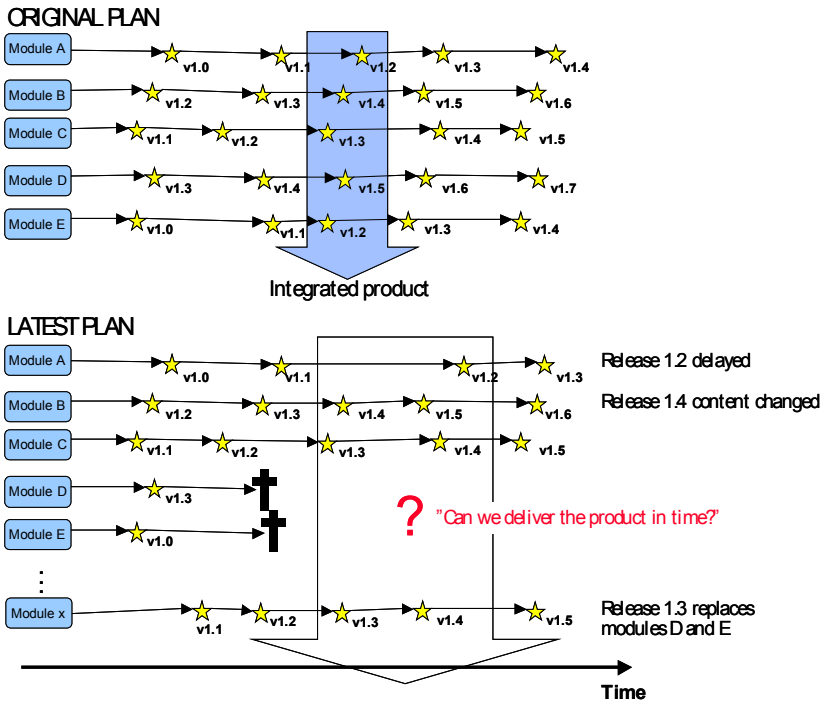

Figure 3. Synchronizing and re-planning of releases to enable deliveries to customer as agreed

Also “backwards-mapping” is important. Situation were everything is compatible with anything is unfortunately not possible in reality. Therefore, a module available sets restrictions to development of new modules – not even talking about restrictions to architecture developments. Thus the life span of a module should be restricted so, that after a certain day, the module is no longer part of the platform. If situation has been changed and the module could still be useful, it can be re-released. According good practice the module is reviewed and need of updating is considered.

5 Conclusion

Main pressure for changes can be allocated to the module properties and to its behaviour. Key issue is the lifecycle of the module; when is it available and for how long? New technology as

a strategic or competitive reason imposes changes to the module and occasionally to the module structure, too. The cumulative effect of increased interactions increases iterations considerably in the product structure development.

It is important to realise what is the actual goal of making dynamic platforms. Product development of mechathronic products is complex by nature. This is due the interactions of different domains and need of co-operation between experts from different mindsets and backgrounds. Thus, often modularisation and platform creation is seen as tool to make development work less complex and thereby easier. However, this is erroneous thinking with dynamic platform. Operating with a dynamic platform is management and engineering challenge and instead of making things easier, adds relations and raises the level of needed competence. The sole purpose of dynamic platform is to give cost and time advantage to company that utilises it. This advantage is gained by extended and systematic design re-use.

The complexity in managing dynamic module structure is easy to understand by bearing in mind that several products (tens of products) are affected if the module under change is planned to be reused in those products. Normally disposition of changes is thought to be unidirectional e.g. from design decisions to production etc. However, in long-term platform management the viewpoint should be wider. Not all changes are coming in form of requirements and knowledge how changes in later process affect to requirements is essential to be able make trade-offs in conflicting situations. According to our opinion the knowledge of these mechanisms is the key for managing of the platform life cycle.

References

[1] Riitahuhta A., Andreasen M. M. "Configuration by Modulaisation", Proceedings of NordDesign 98, KTH, Stockholm, 1998, pp.167-176.

[2] Lehtonen T., Juuti T., Pulkkinen A., Riitahuhta A., "Dynamic modularisation – a challenge for design process and product architecture", Proceedings of International Conferenrence of Engineering Design ICED03, Stockholm 2003.

[3] Laukkanen M. "Conducting causal mapping research: opportunities and challenges", Eden C, Spender J-C, eds. Managerial and Organizational Cognition: Theory, Methods and Research, London: SAGE, 1998, p. 168-191.

[4] Olesen J., "Concurrent Development in Manufacturing – based on dispositional mechanisms", Institute of Engineering Design, Lyngby 1992.

[5] Lindh B-E., "Scania and it's vehicles 1891-1991", Streiffert & Co, Stockholm, 1992.

Corresponding author:

Timo Lehtonen

Product Development Laboratory

Tampere University of Technology

PL 589

FIN – 33101 Tampere, Finland

Tel: Int +358 3 3115 2627

E-mail: timo.lehtonen@tut.fi