On Families of Systems Architecture

Julien Cazalas, Jérôme Gui, Daniel Krob, Loïc Le Sauce

2023 Conference on Systems Engineering Research

On Families of Systems Architecture

Julien Cazalas[a], Jérôme Gui[a], Daniel Krob[a,b], Loïc Le Sauce[a]*

[a] *Center of Excellence on Systems Architecture, Management, Economy & Strategy (CESAMES), 10 rue de Penthièvre, Paris 75008, France*
[b] *INCOSE Fellow*

Abstract

Architected families of systems (or equivalently product lines) are key industrial assets: they enable companies implementing this approach both to differentiate their products and address fragmented markets, but also to pool common elements and thus reduce cost & time-to-market of their products. This paper presents a structured methodology for architecting families of systems, valid for systems in general (i.e. hybrid multi-physical systems mixing software and hardware dimensions); and enabling, via relevant variability indicators, to evaluate and optimize the variability of a family of systems. This paper also presents a variant of this methodology for high-volume families of systems, based on statistical analyses. For these two methodological variants, we present successful actual deployment results on aeronautical, defense and high-tech case studies.

Keywords: Abstraction ; Family of systems ; Feature modelling ; Product line, System ; Systems architecture

We would like to thank the many CESAMES' collaborators who contributed to the various industrial case studies that supported our work: Sevag AINEJIAN, Augustin CURLIER and Antoine MOUTTAPA

1. Introduction

Companies in the industrial, software or service domains are today faced with increasingly fragmented markets where each market segment is characterized, on the one hand, by a high level of specificity expected by customers on functionality, performance, look & feel, etc. of each product and, on the other hand, by increasingly severe competition. In this context, the value of any product or service is linked to its ability to be adapted in order to precisely meet each customer need. This leads to a difficult industrial challenge: the introduction of diversity indeed often

* Corresponding author. Tel.: +33 6 98 22 06 25;
E-mail address: loic.lesauce@cesames.net

increases costs and delays in design, manufacturing, sales and support; so, how can we succeed in industrially producing products that must be increasingly adapted to each customer, i.e. *how can we industrialize tailor-made products*?

The objective of a *product line* – this term designating a family of products, grouped according to one or more common criteria – is to design & manufacture highly customizable products meeting customer needs, in order to differentiate from the competition, while remaining competitive in terms of cost and time to market. Behind this industrial and economic problem lies a scientific problem which boils down to the following statement : on the one hand, how to model a *family of systems* (we prefer to use this term to begin to abstract the underlying business problem) and, on the other hand, how to build a modelling & design methodology adapted to families of systems, based on rigorous principles.

The challenges of such an industrial and scientific problem are huge because most of complex systems manufacturers (e.g. in aeronautics, automotive, defense, energy, railway, etc.) face difficult issues in managing the diversity of their product families and its consequences in terms of recurring & non-recurring cost, time-to-market, etc., which are becoming more and more significant in a context of globalization and exacerbated competition. This problem has therefore already generated a dedicated research stream which is structured around the following main research themes:

Modeling of software system families: the corresponding literature focuses on (1) formalizing the distinction between domain & application engineering [19], (2) defining reusable modular software architecture patterns such as the MVC (Model–View–Controller) or the micro-services architecture patterns [5], (3) designing automatized software configurators based on the concept of "feature", which refers to a property visible from the end-users (see [1] and [13] for more details) and (4) describing good practices of software development organizations operating in a product family mode, based essentially on empirical observations [5]. This research topic is rich and active, but we can nevertheless see that it is only limited to families of software-dominant systems, excluding general hybrid multi-physical systems. Due to the structural differences between software systems and systems in full generality (software systems being only a “simple” particular case from the point of view of general systems theory), it is notably impossible to reuse this work identically when we approach the design of general families of systems.

Modularity of general families of systems, i.e., multi-physical hybrid systems mixing software and hardware dimensions: the existing literature focuses here on specific themes, such as (1) modularity and decoupling between the constituent subsystems of a family of systems: in this respect, we can especially cite [2], [3] and [23], which endeavored to define interface patterns between modules, but also the conceptualization of the notion of modularity, based on Design Structure Matrices (DSM), as defined in [14] or [20], (2) optimization of system families (see for instance [8], [16], [21] or [22]), (3) design of manufacturing systems dedicated to families of systems, using techniques such as delayed differentiation and mass customization, taking into account the impacts of a product line in terms of industrial organization and cost (see [10], [11], [12], [17] or [22]) and (4) study & analysis of ad hoc industrial cases (see [4], [6], [9] or [18]) which do present industrial examples of deployment of product families, but without identifying generic methodologies.

As a consequence, we did not really find in the literature – to the best of our knowledge – a structured method for designing general families of systems. Such a methodology shall ideally be end-to-end, generic and applicable to all types – both multi-physical and software – families of systems, based on indicators for measuring variability, and cover as well the scope of large-scale systems that pose specific problems of modular statistical analysis.

Note finally that we constructed five years ago an initial version of a families of systems architecture methodology, that relied on a classical top-down system approach: identifying stakeholders needs, then deriving system functions & requirements, and then defining the modular architecture of the family of systems. This approach was tested in the field and prove to be a failure, due to combinatorial explosion of manipulated concepts and the lack of applicability of certain proposed concepts. The central objective of our paper is to present a reconceptualized version of this methodology

2. Our systems architecture framework for families of systems

2.1. Our conceptual framework for families of systems architecture

Let us first present the general framework in which our work is integrated. To this end, we shall recall that a *formal system* S is fundamentally defined by sets of inputs, outputs and states, a time scale and two evolution functional equations expressing the fact that the system S produces an output and changes of state under a given input at each moment within its time scale (see [15] for more details).

Relying on Cousot and Cousot's theory of abstract interpretation (cf. [7]), we can then formally model a *family of systems*, by defining the notion of family of formal systems as follows:

Definition – Families of formal systems: Let $F = (S_i)_{i=1..N}$ a set of formal systems (in the sense of the CESAM framework; see [15]). Then we will say that F is a *family of formal systems*, as soon as one can find a formal system forming an abstraction, in the sense of abstract interpretation (cf. [7]), of all the systems of the considered family F.

In other words, F is a family of formal systems if there is an abstraction application α (in the meaning of abstract interpretation, [7]) sending each system S_i of the family F to a single system S that abstracts all systems of F and if there is also a concretization application χ_i (again, in the meaning of abstract interpretation, [7]) to obtain each system S_i from S (this mechanism captures the pragmatic notion of product configuration). In such an approach, the difficulty is twofold since one needs 1) to find the right abstraction-concretization mechanisms to model a given family of systems, 2) to adapt a systems architecture process to the system that abstracts all the systems of the considered family of systems.

One of the key points of our approach is also to place ourselves in a conceptual framework where the abstractions used to represent families of systems have an architectural meaning, in this case based on a flexible modular

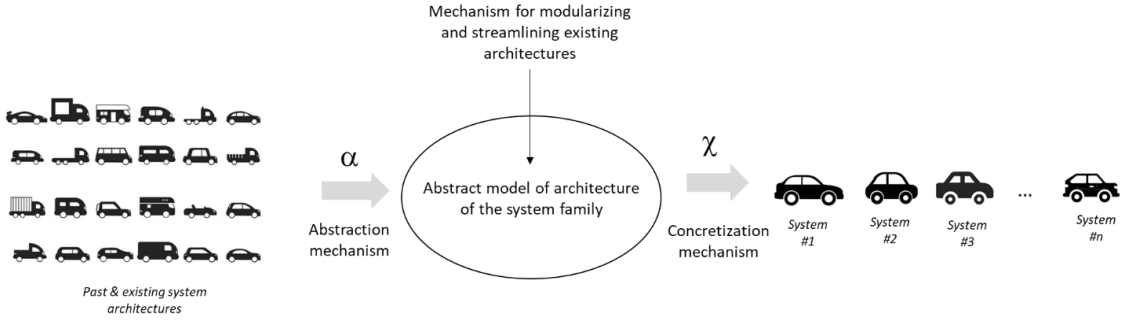


Figure 1 – The notion of family of system

architecture approach. This consists in considering that a family of systems is always built on top of a platform formed by a set of components that are common to all the systems of the family (in green in Figure 2). A system of the family is then obtained by adding to this platform a first set of standard modules (in yellow in Figure 2), that are themselves still common to a subfamily of the family considered, then finally a set of completely specific modules (in red in Figure 2). This gives raise to the symbolic representation of the entire family of systems provided in Figure 2.

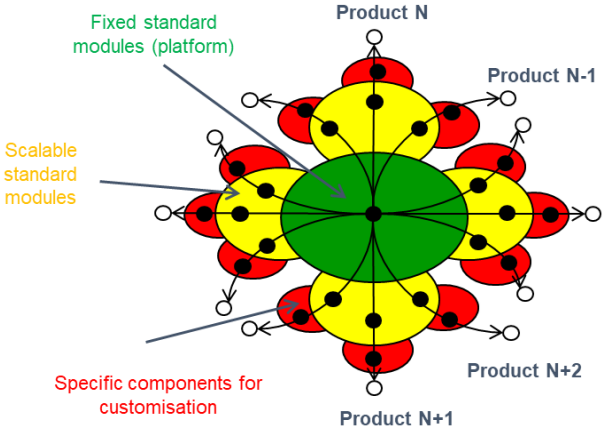


Figure 2 – Symbolic representation of a family of system

Most industrial systems managed in diversity are made according to such an architectural pattern: almost all automotive vehicles are now built according to such a logic with a platform common to many types of cars, then a first set of modules to obtain a given type of silhouette and finally a set of customized modules bearing the final differentiation offered to the customer. Many consumer products – ranging from do-it-yourself devices to yogurts – are also built according to the same principle.

2.2. Our systems architecture conceptual framework for families of systems

We begin here by quickly presenting the core principles that guide our framework dedicated to the architecture of families of systems.

2.2.1. Principle 1: Analyze the modeling of families of systems as a process of modularization and rationalization of existing systems architectures

As a first structuring principle, our systems architecture framework for families of systems relies on a *bottom-up approach* (contrary to our first method, founded on a needs-driven top-down approach), fundamentally based on modularizing and rationalizing a set of existing systems architectures that form the core input of our systems architecture method for systems families, as illustrated in Figure 1.

2.2.2. Principle n°2: Position the product breakdown structure as a pivot of the abstract modeling of a family of systems

Let us define the logical breakdown of a family of systems – called Product Breakdown Structure 150 % (PBS 150 %) – as an abstract breakdown that provides the hierarchical identification and structuring of all the components necessary to generate all the products of the considered family of systems. Note that this logical breakdown is not the breakdown of a particular system, but an abstract breakdown that works for all the systems of a given family of systems (see Figure 7 and Figure 8 for examples).

Our second principle positions then the PBS 150 % as the central structure around which all abstract design actions, such as variability analyses or statistical analyses, relatively to a family of systems will be carried out, as presented more in details in the sequel of this paper.

2.2.3. Principle n°3: Assess the variability of a family of systems with a standard indicator

Our third key principle is to *assess the variability of a family of systems* with a standard indicator which measures the level of variability of each abstract component of a given family of systems according to the analysis grid that consists in classifying each such system component involved within the considered family of systems into a standard, variant discrete, variant continuous or customized component, as proposed in Figure 3. This indicator makes it possible to quickly assess the variability of an existing family of systems, i.e. typically the design heritage of an enterprise, and to measure the variability of a target family of systems.

The component is …	Variability level	Optional
Standard: same of all systems of the family	S	O
Variant discrete: a finite **number of components** are present in the systems family	VD	
Variant continuous: the component is generated by parametrization	VC	O
Customised: The component is specifically generated (ad hoc) for each system in the family	C	

In addition to (all) level of variability, components may be optional or mandatory

Figure 3 – Variability assessment indicators

2.2.4. The meta-model of our family of systems architecture framework

Finally, the meta-model of our families of systems architecture framework is presented in Figure 4. It shows the key concepts of our methodology, including elements external to a given family of systems (environment, stakeholders, needs, market segments, etc.), elements associated with the internal architecture of the family of systems (PBS, modules, components, etc.), elements relating to variability modeling (variant drivers, features, variability

assessment, etc.) and finally elements relating to the corresponding industrial system modeling (assembly, workstation, etc.).

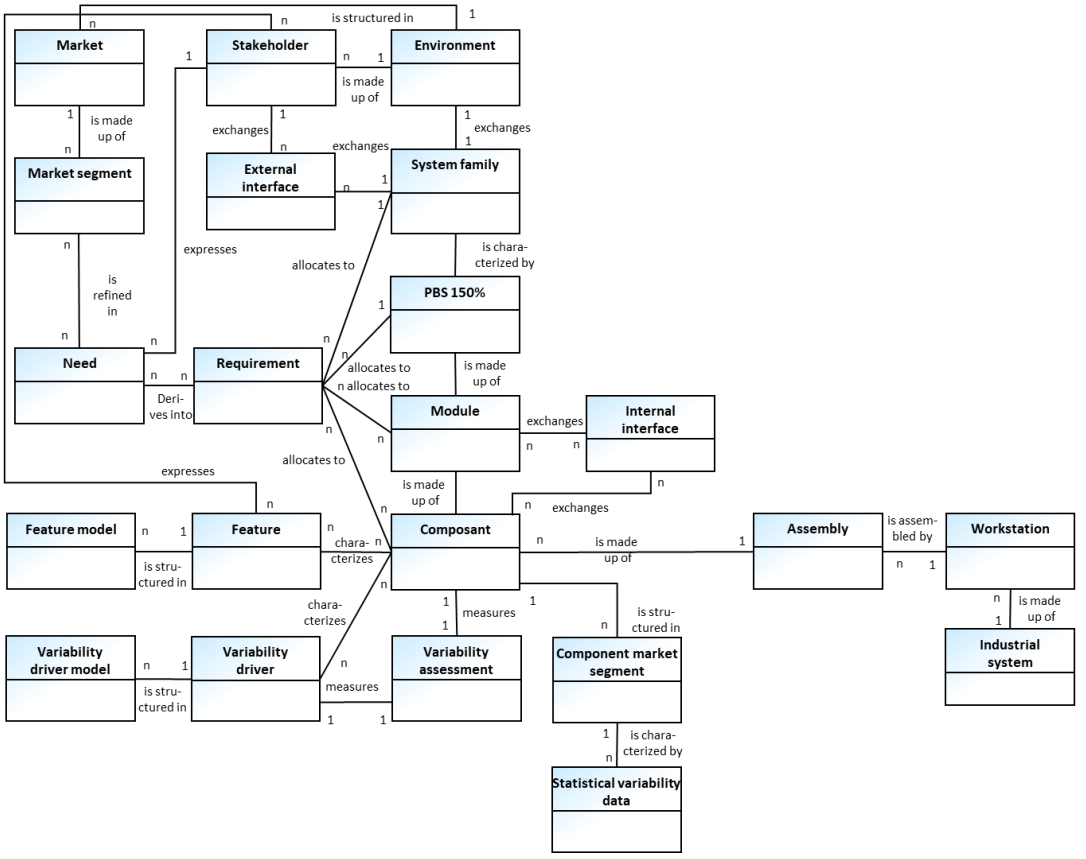


Figure 4 – The meta-model of our framework for families of systems architecture

3. Our methodology for architecting a family of systems

The methodology for architecting a family of systems, presented in this section, was experimentally prototyped & validated on two industrial projects: (1) a commercial aircraft power and data distribution system (DEWIS system), (2) a software-intensive defense system (mission preparation & control system). On case (2), the deployment of our methodology especially allowed to increase the reuse rate of "off-the-shelf" modules from approximately 7 % to 91 %

Our core result is therefore a field-robust methodology for modeling families of systems, fully aligned with the framework presented in section 2. It consists in five main steps, as depicted in Figure 5, that are described more in details here below.

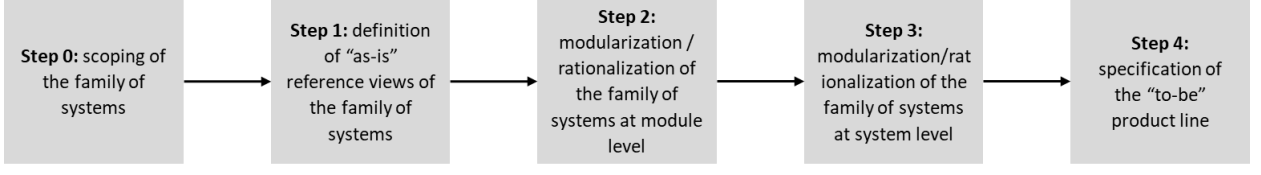


Figure 5 – Main steps of our methodology for modeling families of systems

3.1. Step 0: scoping of the family of systems

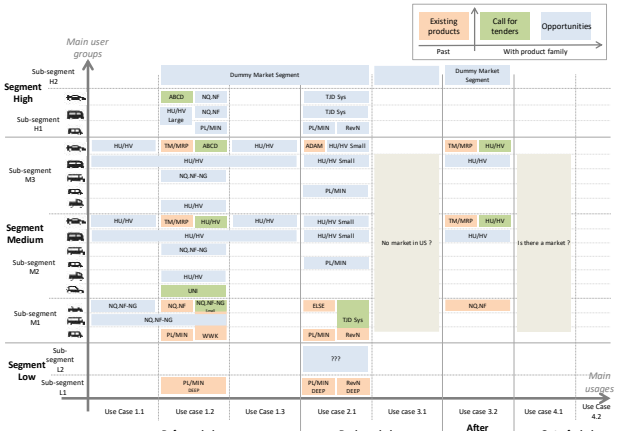


Figure 6 – Example of market segmentation addressed by a family of systems

An initial scoping step consists in clarifying the scope of the considered system family (see also [18]), both from a technical point of view (elicitation of the set of concerned components) and from a market point of view (elicitation of the concerned market segments). The aim of this first fundamental step is to define as precisely as possible which family of systems is in question, in order to avoid any ambiguity on this subject later on. We refer to Figure 7 for a concrete example of market segmentation.

This step also has the role of defining the main indicators to measure the effectiveness of the family of systems in the long term. These include of course the variability indicators as defined in section 2, but also other business indicators such as cost and time-to-market indicators with respect to a particular system instance or to the business model of the family of systems.

3.2. Step 1: definition of "as-is" reference views of the family of systems

The objective of step 1 is then to define a series of systems architecture reference views, representing the current state of the system family. The first of these views is, of course, the PBS 150 % (abstract logical breakdown of the family of systems) that we already introduced. For example, we give in the below Figure 7 an example of a 150 % PBS of an actual family of systems coming from one of the two industrial case studies on which our approach was prototyped. At this stage, we shall also define views derived from these fundamental views: the logical interaction view (representing in particular the interfaces between the abstract components of a family of systems), the geometric

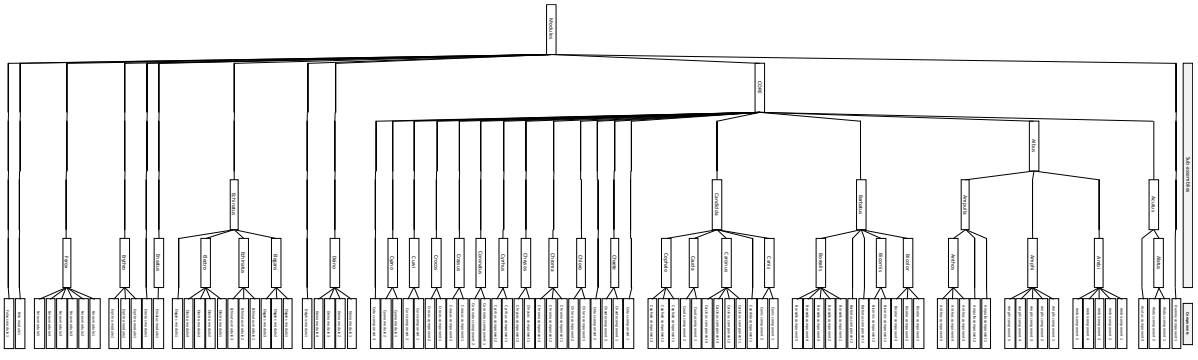
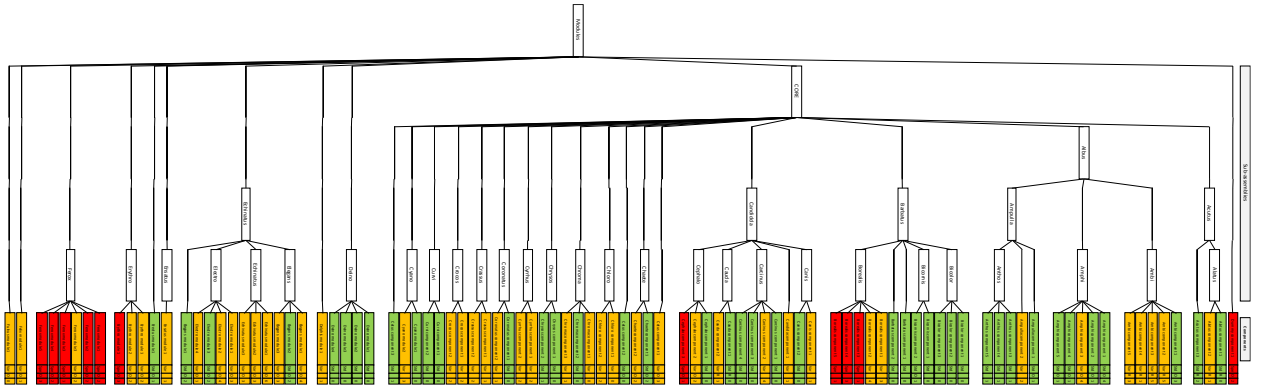


Figure 7 – Example of a 150 % Product Breakdown Structure of a system family

view (3D view), as well as the enterprise architecture of the associated industrial system (if the latter is within the scope of interest of the system family architecture approach, i.e. if it is impacted by it).

3.3. Step 2: modularization / streamlining of the family of systems at module level

The objective of the next step 2 – which is the core of our methodology – is to conduct modularization analyses in order to streamline the family of systems in question. This is done by assessing the initial variability ("as-is") and defining the target variability objectives ("to-be"), using the variability assessment indicator that we defined in Section 2. The below Figure 8 shows for instance a 150 % PBS after variability streamlining and the associated variability measurement.



Modules variability		
	As-is	To-be
Number of modules	59	90
Standard	5%	57%
Variant Continuous	2%	34%
Customized	93%	9%

Figure 8 – 150 % PBS after variability streamlining

We also rely on an original technique that we developed, that is to say the analysis of *variability drivers*, consisting of systematically evaluating the root causes of variability of an abstract component, in order to separate the "good" causes (related to customer value, for example) from the "bad" causes (related to an internal increase of complexity, for example) of variability, in order to eradicate variability not related to a "good" origin. Some of the techniques used in this step focus on the modularization of the target modular architecture (see [2]) through the use of modular architecture patterns (see [5]) and the definition of standardized interfaces. Finally, we also integrated statistical analyses, when relevant, in this step, which we will present in more detail in section 4.

3.4. Step 3: modularization / streamlining of the family of systems at system level

This step in the methodology of architecting a family of systems deals with the process of instantiation of the family of systems, i.e. the process of concretization (as defined above and illustrated in Figure 1) in the sense of abstract interpretation [7]. This mechanism is based on an architecture view, called a *feature model* (see [1] and [13]) which formally corresponds to a needs to abstract modules matrix.

The "feature model" is the support of the concretization mechanism, in the sense that it works as a decision tree allowing the selection of a set of properties corresponding to a set of requirements (the *features*). This requirements

to abstract modules matrix is part of the overall model of the system family and captures its variability in order to allow the selection of abstract modular components and their performance (see Figure 9 for an example).

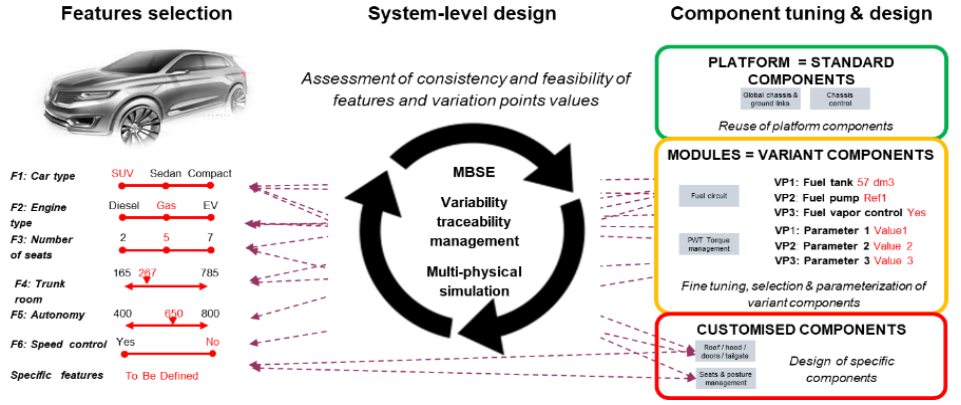


Figure 9 – Principle of a feature model

3.5. Step 4: specification of the "to-be" product line

This fourth and last step produces the specification of the different modules of the target system family by identifying the system-level requirements, and deriving them at the module and component levels, in order to specify the performance ranges of the various components & parts that make up the system family. In the special case of manufactured systems, the modelling and potential re-arrangement of the assembly line is dealt with at this step, following a delayed differentiation principle (see [10] and [11]).

4. The special case of large volume families: a statistical approach

This final section deals with the definition of a statistical methodology dedicated to the specific case of families of large series of systems whose interfaces are highly standardized. This particular type of families of systems does not require architecture modularization, but rather variability streamlining based on statistical data analysis. This methodology was deployed in a high-tech context and enabled a reduction in the number of references on certain key components from 50 % to 83 %.

4.1. Methodology presentation

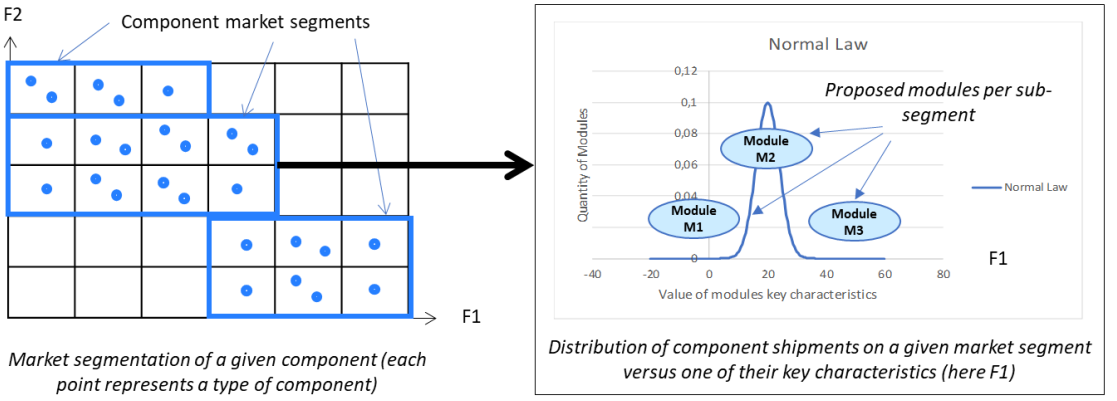


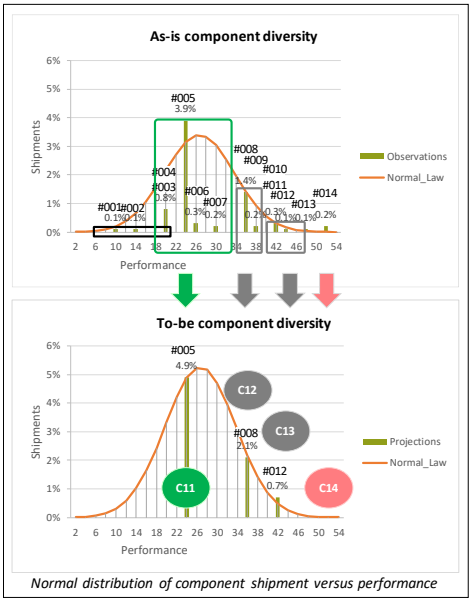
Figure 10 – Overall view of our statistical approach for large volume product families

This method is deployed in two steps as depicted in Figure 10. The first step consists in (1) Identifying the two main characteristics F1 and F2 (in a principal component analysis logic) that characterize the technical component under consideration, (2) breaking down the component market into a series of homogeneous market segments described in terms of F1 and F2 characteristics, and then (3) verifying that the market segments that were identified in the previous step are homogeneous (i.e. that the sales distributions on these segments typically follow Normal distributions according to the most structuring F1 or F2 characteristics).

The second step consists then in identifying a number of *reference modules* that do cover well the considered component segment of market: this can for example be done by taking the existing modules that are the most representative in terms of low, medium and high performances with respect to F1 or F2 characteristics. Classical statistical tests (e.g. χ^2) are used to strengthen the analysis of market homogeneity and the choice of reference modules. This second step is crucial in order to remove unnecessary components that cover the same component market segment and thus streamline the components diversity of the product family. The key idea is therefore to be able to cover the complete component market, but only using a limited number of reference modules.

4.2. A concrete deployment example

The method we have just presented was applied to the analysis of a family of high-volume high-tech products (see Figure 11 for an example of the application of this method to the modular analysis of a component for various segments of market). It should be noted that the deployment of our method in this situation confronted us with a mathematical problem, namely the identification of the Normal distribution that best approximates a set of data, which was solved using optimization techniques.



Component	As-is	Possible to-be	Diversity reduction
Segment 1	35 different references	11 standard modules	– 68.5 %
Segment 2	26 different references	14 standard modules	– 53.8 %
Segment 3	7 different references	3 standard modules	– 57 %
Segment 4	6 different references	3 standard modules	– 50 %
Segment 5	24 different references	4 standard modules	– 83.3 %

Figure 11 – A concrete illustration of our statistical methodology for large volume product families

In the context of this latest case study, we were also interested in better understanding the impact of modular approaches on manufacturing from several angles: optimizing product definition via product configuration automation, anticipating sales figures via capacity planning techniques, and optimizing standardized module testing strategies, which can now be achieved based on the underlying modular architecture that easily supports such optimizations.

4.3. A step further: development of a product configurator

This last result has also paved the way for the realization of a system family configurator, also called a product configurator, implementing the concretization process which is a the heart of our definition of a family of systems and of the "feature model" approach that is part of the step 3.4 above, as already described previously.

Such a configurator takes the values of the characteristics of a target system within a family of systems that are desired by a customer, and then automatically produces the technical configuration of the system that meets these characteristics, based on a reference modular architecture. The underlying mechanism of course uses a "desired characteristics to produced modules" correspondence table, which needs to be made explicit a priori. Here, we have worked to bring out the functional architecture of such a configurator, and to validate this architecture via rapid - and disposable - prototyping, the sole purpose of which is to serve as a proof of concept. Figure 12 illustrates the work accomplished in this respect. It is worth noting that this last activity validates the fact that such an approach allows to automate part of the engineering work, as long as an underlying modular architecture is used, and thus materializes the gains of this approach.

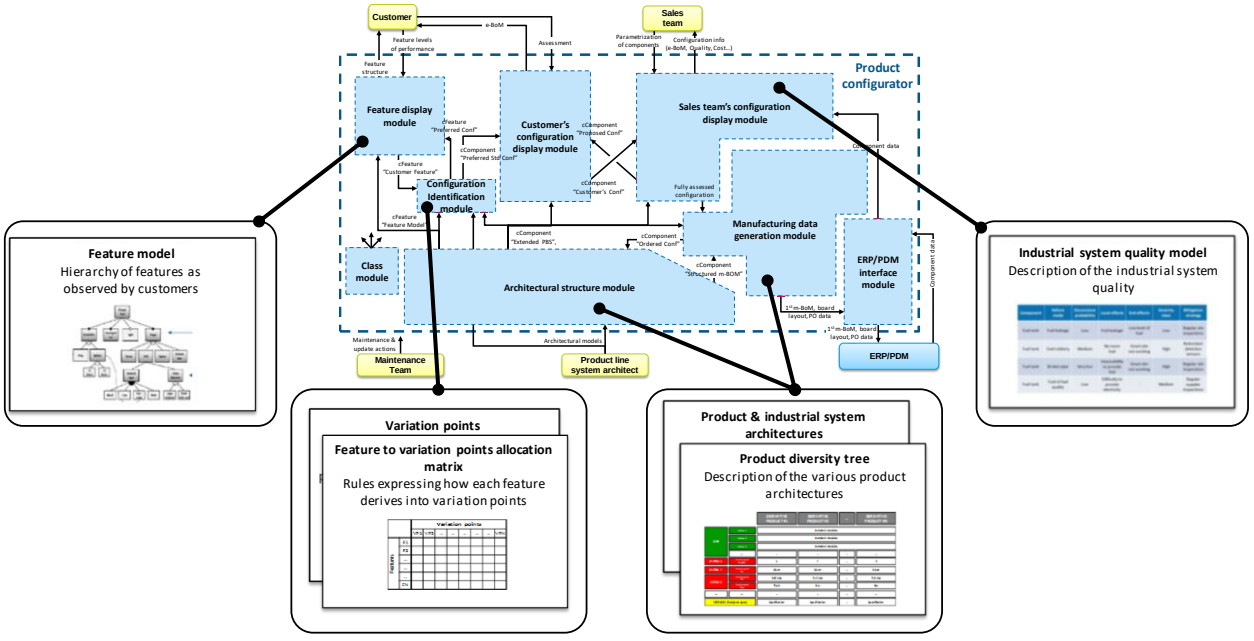


Figure 12 – Simplified functional architecture of a system family configurator

5. Conclusion

In this paper, we presented a proposed framework and structured methodology for systems family architecture, enabling the modeling and optimization of systems family variability in general, as well as a variant of this methodology applicable to large volume product families. The deployment of this methodology on concrete application cases has shown that this framework and methodology can be used to pragmatically model systems families in a variety of industrial sectors, and to achieve significant reductions in variability on the concerned systems families.

Nevertheless, this paper is only an intermediate step. Future work on this framework and methodology should focus on the notion of modularization of the systems families concerned, through the modeling and definition of interface patterns enabling modularity, as well as on the tooling of the methodology, for example through extensions to SysML modelling tools.

Bibliography

[1] Apel S., Batory D., Kästner C., Saake G., *Feature-oriented software product lines*, Springer, 2016

[2] Baldwin C. Y., Clark K. B.. *Design rules: The power of modularity (Vol. 1).*, MIT press., 2000

[3] Baldwin C. Y., Clark K. B., *The option value of modularity in design*, Harvard NOM Research Paper, 1, 1-14. 2002

[4] Clements P., Bergey J., *The U.S. Army's Common Avionics Architecture System (CAAS) Product Line: A Case Study*, Technical Report CMU/SEI-2005-TR-019, 2005

[5] Clements P., Northrop L., *Software Product Lines: Practices and Patterns*, Addison-Wesley Professional Ed. ISBN 0-201-70332-7, 2001

[6] Cohen S., Dunn E., Soule A., *Successful Product Line Development and Sustainment: A DoD Case Study*, CMU/SEI-2002-TN-018, 2002

[7] Cousot P., *Interprétation abstraite*, TSI, Vol. 19, n°1, p. 1-9, Hermès, 2000

[8] Deng S., Aydin R., Kwong C. K., & Huang, Y., *Integrated product line design and supplier selection: A multi-objective optimization paradigm*, Computers & Industrial Engineering, 70, 150-158., 2014

[9] Guertin, N., and Clements, P., *Comparing Acquisition Strategies: Open Architecture vs. Product Lines*, Proceedings of the 2010 Acquisition Research Symposium, Monterey, 2010

[10] He D., Kusiak A., *Design of assembly systems for modular products* , IEEE Transactions on Robotics and Automation, vol. 13, no 5, p. 646-655, 1997

[11] He D., Kusiak A., Tseng T., *Delayed product differentiation : a design and manufacturing perspective*, Computer-Aided Design, vol. 30, no 2, p. 105-113, 1998

[12] JiaoJ., Tseng M., A *methodology of developing product family architecture for mass customization*, Journal of Intelligent Manufacturing, vol. 10, p. 3-20., 1999

[13] Kang K. C., Lee J, Donohoe, P.. *Feature-oriented product line engineering.*, IEEE software, 19(4), 58-65, 2022

[14] Kossiakoff A., Sweet W.N., *Systems engineering – Principles and practice*, Wiley, 2003

[15] Krob D., *Model-Based Systems Architecting – Using CESAM to Architect Complex Systems*, ISTE, Wiley, 2022

[16] Kwong C. K., Luo, X. G., Tang, J. F. *A multiobjective optimization approach for product line design*, IEEE Transactions on Engineering Management, 58(1), 97-108, 2010

[17] Lee H., Tang C., *Modelling the costs and benefits of delayed product differentiation*, Management Science, vol. 43, no 1, p. 40-53., 1997

[18] Lehnerd A. P., Meyer M. H, *The Power of Product Platforms*. Simon and Schuster, 2011

[19] Pohl K., Böckle G., Linden F., *Software Product Line Engineering, Foundations, Principles and Techniques*, Springer, 2005

[20] Sage A.P., Armstrong J.E. Jr., *Introduction to systems engineering*, John Wiley, 2000

[21] Simpson T.W., Siddique Z., Jiao J.R., Eds, *Product platform and product design*, Springer, 2006

[22] Simpson T.W., Siddique Z., Jiao J.R, *Product family design and platform-based product development: a state-of-the-art review*, Journal of intelligent Manufacturing, (18), 5-29, Springer, 2007

[23] Ulrich, K., *The role of product architecture in the manufacturing firm*, Research Policy, Volume 24, Issue 3, Pages 419-440, 1995