Supporting Evolving Product Families

Piërre van de Laar

Embedded Systems Institute, The Netherlands, pierre.van.de.laar@esi.nl

Abstract

The goal of this article is to draw attention to the challenging problems associated with supporting evolving product families. After a general problem description, we focus on a single detail of supporting evolving product families. We propose and evaluate using industrial experts a method to measure the similarity between products.

Key words – product family, evolution, evolvability, service, upgrade, test, installed base.

1 Introduction

Customers not only want more and more personalized products, they also want to keep their products more and more up-to-date. For example, when buying a car, customers will select the features they like from the feature list, containing, for example, car radio, navigation system, DVD player, and cruise control. Later, customers want to listen to the music on their iPod in the car; have the latest maps for their navigation system; watch blue ray disks next to DVDs; have adaptive cruise control; and add both a hold-your-line and a parking assistant.

The requirement of more and more personalized products is addressed extensively by product families [4][15][18][20]. The requirement of keeping these products up-to-date has received only very limited attention [14]. We see two reasons for this limited attention:

- Supporting product families is even more complex than developing them, since whereas development only addresses the products on sale in the next period, support also addresses all products sold.
- The relevance of support depends on the kind of product. Many consumer devices, like electronic toys, mobile phones and televisions, are considered throw-away devices. These devices are typically not serviced and not upgraded: they are simply replaced.

With respect to the last reason, we see a change currently taking place: Consumer devices are becoming more and more upgradeable. For example, the latest mobile phones can be personalized using ring tones, skins, and even applications; and the newest televisions are made upgradeable to mitigate the increase in risks associated with going from a stand-alone device to a node in the network.

Product families realise mass customisation [17] by balancing the variety in products with the costs involved. This balance will be discussed more thoroughly in section 2. In section 3, we focus on a number of questions that typically must be addressed while supporting evolving product families. From these questions, we extract our research question that addresses a single detail of supporting evolving product families. In section 4, we discuss related work. In section 5, we describe the solution direction we investigated. We end in sections 6, 7, and 8, with a summary, a discussion, and the road ahead of us.

2 Product families and similarity

Product family development focuses on multiple products. Trade-offs are made in the context of these products. The commonalities between product variations and/or product generations are exploited. Products are composed out of smaller parts, possibly in multiple steps. These parts are developed for usage in different products. Multiple products are constructed using the same parts. The products in a product family developed by an organisation are dependent on each other.

Customers want mass customisation: a large variety of products for a reasonable price. Product families realise mass customisation [17] by balancing the variety in products with the costs involved. The costs are related to development, testing, upgrading, maintenance, servicing, and bill of materials:

- The larger the differences between products in a product family, the more complex the development and testing becomes. When complexity increases, the costs and time to market increases as well.
- The more products are similar in the installed base, the easier it becomes to make upgrades for the installed base. Furthermore, the same upgrade will address a larger potential customer base.
- The more products are similar in the installed base, the cheaper their maintenance and servicing. For maintenance, not only fewer different spare parts are needed but also more products depend on the remaining spare parts. When the amount of requests for a spare part increases, these requests become better predictable due to the law of large numbers. As a consequence, the number of items on stock can be optimized even better. In addition, service engineers not only need less education, there is also less need for specialization, which makes assigning service engineers to the products needing service easier.
- The more similar products are, the more they profit from the economy of scale of their constituent parts.

To make the right trade-off in the variety supported by a product family, we need to balance the value of the differences between products with the costs to realize them.

3 Supporting evolving product families

To understand the issues of supporting evolving product families, we first describe the support of a product instance during its lifecycle. Second, we look at the reasons why product families evolve. Third, we focus on a few questions that must be answered while supporting evolving product families. From these questions, we extract our research question.

3.1 The lifecycle of a product

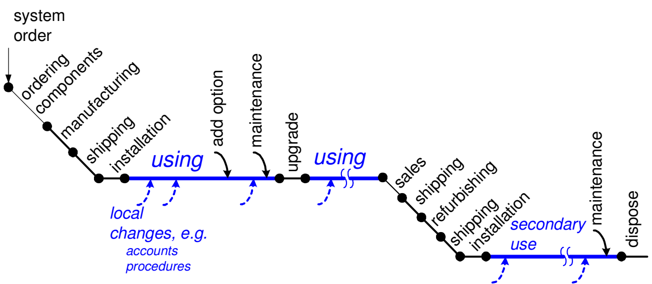


Figure 1 - *The lifecycle of an industrial product [10]***.**

To illustrate the support that a product instance receives during its lifecycle, we describe the life of an industrial product, such as an MRI scanner, as depicted in Figure 1. The life of a product instance starts when an order for it is placed. The product is constructed according to the customer's wishes. Part of the construction happens in the factory; part at the customer's site. Once constructed, the product is ready to be used. The product is configurable, e.g., to handle accounts to ensure the privacy of patients' data, and its configuration settings will be changed during its lifetime. The product is kept up-to-date by options and upgrades, which might require that the product is temporarily shut-down. Since the performance of the product deteriorates over time, e.g., due to wear, service at regular points in time and occasionally, in cases of break down, is needed. When the product does not any longer satisfy the needs of the customer, the product can be sold to another user, directly or indirectly via a broker. In the latter case, the product is often refurbished to better meet customer's needs, before installing it at the site of the second-hand buyer. The cycle of usage and re-selling ends when the product is finally disposed and recycled.

3.2 Why do product families evolve over time?

Product families evolve to follow changes in technology, environment, and stakeholder's needs [12]. To give some MRI scanner related examples:

- Technological advancements in the computing infrastructure, such as faster processors, larger memories, and 64 bits operating systems, enabled handling of even larger clinical images. Technological advancements in MRI lead to new clinical applications, such as functional MRI.
- New legislation, such as the European Physical Agents Directive, could dramatically change the working practices of MRI personnel.
- The unsatisfied need of surgeons to know where fibres are positioned inside the brain to prevent accidentally cutting them during surgery, stimulated advancements in the product's domain leading to a new clinical application called fibre tracking.

While evolving product families, many companies have the strategy to make backwards compatible changes. In other words, these companies try to ensure that products in the installed base can be upgraded to provide the same functionality as the products on sale. Especially when the changes have localized impacts, backwards compatible changes can often be realized.

3.3 Extracting our research question

Many industrial questions are raised, while supporting evolving product families. In this section, we focus on five of these questions, from which we will extract our research question that addresses a single detail of supporting evolving product families.

3.3.1 Which products to test for a replacement?

Products contain hundreds of parts. Many of these parts are bought from an external provider. Buying parts from a provider enables a company to focus on its own core competencies while benefiting from the provider's expertise and economy of scale. However, these parts are not under the control of the company making the complete product. Hence, they can become obsolete, for example, when a last-time-buy call is issued by the external provider or due to bankruptcy of the provider. When parts are no longer available, a replacement is needed. Typically, tests will be performed to ensure that a replacement is indeed compatible in form, fit, and function. Since a part can be used[1] in many products, many different tests are possible. While tests reduce the risk associated with using the replacement, costs are associated with performing them. Therefore, companies want to perform tests on that set of products that optimally balances the cost of performing these tests and the risk associated with using the replacement for the installed base. Tests are, however, not equally effective in reducing that risk. At least, two rules determine the effectively of a test. First, the more instances of a product are in the installed base, the more effective a test of that product is. Second, the more a product differs from the products used in earlier tests, the more effective a test with that product is. The first rule raises a relative simple question that a company typically can answer: which products are in the installed base? The second rule raises another question: how can the difference between products be measured?

3.3.2 How to profit maximally from supporting the installed base?

Many companies earn more money with supporting their products than with actually selling them. Well-known

[1] A part does not have to be contained in a configuration to be used: a part is used when it is a replacement for another, still functioning but no longer available, part.

examples are game consoles and printers, where the profit is made on the games and ink, respectively. Other sources of revenues generated by the installed base include service contracts and selling of upgrades. Typically, spare parts are needed for them, i.e., to be able to replace a broken part or to realize the upgrade. Costs are, of course, associated with maintaining a stock of spare parts. The more different parts are contained in the products being serviced, the higher the costs associated with storing their spare parts. When the products in the installed base could be made more similar, costs would also be reduced. This can be stimulated by pricing upgrades attractively for the most different products in the installed base. But how can the similarity and difference between products be measured?

3.3.3 How risky is upgrading a specific product?

Customers of a product typically want to keep it up-to-date. Especially when the substantial investments are involved, upgrades are a viable option to achieve this goal. To give some car-related upgrade examples:

- A FM car radio can be replaced by a digital one which can play CDs and mp3s as well;
- Car navigation can be added; and
- Cruise control can be built in.

The first two examples are typically done by the customer himself. The last example is however typically done by a service organization: the garage. An upgrade often targets multiple products. For example, the first two upgrade examples are typically independent of the car's brand. The chance that an upgrade targets multiple products is even higher when the products are members of the same product family. Before an upgrade is offered to customers, tests using a subset of the targeted products are typically performed. This subset is selected to balance the risk of applying the upgrade and the test costs involved. Still when an upgrade is sold to a particular customer, a risk assessment is important, especially when the upgrade is performed by a service organization. This service organization needs among others to communicate the appropriate amount of time needed to perform the upgrade, and to assign the upgrade task to either a local service engineer or the upgrade's expert. The more different the product of the customer is from the products used in the tests (and earlier successful upgrades), the larger the risk of applying the upgrade. But how can the difference between products be measured?

3.3.4 Which products should be available for test purposes?

Customers of products might experience problems. To analyse these so called field problem reports, a duplicate of each released product could be kept. And to analyse the field problem reports quickly, a duplicate of each released product should even be available, i.e., installed and fully functional, since building a product can be quite time consuming: building an MRI scanner takes, for example, one week. Keeping a duplicate of each released product is costly. Not only due to the costs associated with each product but also due to storage costs. Having a duplicate of each released product available increases the costs even more, since an installed product requires more space than its stored parts and an installed product requires maintenance and service. To balance the costs of having duplicates of released products available (or stored) with the risks associated with field problem reports, one should have duplicates of released products that are maximally similar with the installed base, while being maximally different with each other. But how can the similarity and difference between products be measured?

3.3.5 How to select a pilot site?

New functionality in a product family is often introduced to the market in steps. For example, new functionality can initially only be offered to a limited set of customers: a few pilot sites, or limited to a (small) country. By limiting the number of customers also the risk is limited, while valuable user feedback of the new functionality will still be received. The user feedback is caused not only by the differences in the context of usage, i.e., laboratory versus actual usage, but also by the differences in the products used. Hence, one criterion to select the limited set of customers is to maximise the differences with the products used in the laboratory tests. But how can the difference between products be measured?

3.3.6 Research question

While supporting evolving product families, many industrial questions are raised. Five questions, we addressed in more details, turned out to share at least one underlying question. In industry this question is currently answered using experience and gut feeling. Hence, we consider research to improve this way-of-working valuable for the following reasons:

- Knowledge in the heads of experts is volatile, since experts move to other jobs and retire. Capturing this knowledge makes it less volatile.
- Not only the experts' knowledge is implicit, also inconsistencies between experts remain hidden. By making the knowledge explicit, inconsistencies become clear and can be resolved.
- Whereas the cognitive capacities of experts remain the same, the complexity of many industrial questions increases, e.g., due to an increase in the products sold per year. Hence, the experts need support to be able to answer many industrial questions at least as well and fast as before.

Our research question that we address in the reminder of this paper is: How can the (dis)similarity between products be measured?

4 Related work

To our knowledge, [14] is the only article that explicitly addresses one aspect of supporting evolving product families. [14] proposes to exploit commonalities between products in order to reduce the verification effort. [14] measures the similarity between products using "locality sets" that contain the architectural elements that realise the functionality concerned by a certain requirement.

Verification of that requirement is required to be independent of the behaviour of any architectural element contained in the product but not in the locality set. In the worst case, the locality set contains all architectural elements of the product. Using the locality sets, the set of representative products is determined such that the successful functional verification of this small set implies the functional correctness of the entire product family. We have doubts whether the approach using locality sets can be generalised to also guarantee non-functional correctness, since non-functional dependencies between the functionalities due to, for example, shared buses, memory, and processors, impact many non-functional properties in nontrivial ways.

The field of product family testing [8][16] so far only addressed the testing of products to be released to the customers. Because many test-related questions also relate to the installed base, we expect that product family testing will in the near future also include sold products that still must be supported into the test plans. Product family testing exploits similarities between the products, such as shared requirements expressed as use cases [3][11], the shared architecture [9] and the shared framework [1].

The field of independent lifecycles [7][13] addresses the impact on a product of asynchronous obsolescence of its constituent parts. By developing models to manage the product's evolution, minimal product ownership costs are realized. These models specify when the product should evolve, i.e., when which part is to be replaced by which other part. Our work complements theirs. Whereas they assume that technical feasibility of part replacement has already been achieved, we explicit address the question whether a part is a viable replacement for another part (see section 3.3.1).

5 Solution direction

The research described in this section is executed as part of Darwin [19]. Darwin is a collaborative project between the Embedded Systems Institute, Philips Healthcare, Philips Research, and five Dutch universities (Delft, Eindhoven, Groningen, Twente, and the VU University of Amsterdam). The project started end of 2005 and will run until the end of 2010. The size of the staff of the project is equivalent to 20 full-time people, and includes 10 PhD students and 2 Postdocs. The goal of Darwin is to understand evolvability as a system property; to identify, create, and apply constructs, models, and methods to support evolvability; to support the trade-off decisions the architect will have to make with respect to evolvability; and to support the sub-system and technology lifecycle view of a system. The Darwin project is carried out using the industry-as-laboratory paradigm. Hence, the researchers are working closely together with developers of Philips Healthcare MRI, a large organization that produces MRI scanners: Embedded systems with a lifetime of over a decade, which are used in hospitals to visualize the structure and function of patient's bodies. Furthermore, the researchers have access to a large source of information, including a large archive going back for many years. Within the archive of Philips Healthcare MRI many different databases exist that contain a wealth of information related to products and the parts (both hardware and software) they contain. Unfortunately, each database has its own point of view, e.g., sales, service, or logistics, and each database is maintained in isolation which complicates relating the content in different databases.

In this section, we describe the solution direction we took to answer the question: How can the (dis)similarity between products be measured? Since industrial experts are currently answering this question, we cooperated extensively with experts from Philips Healthcare MRI. At all times, we kept our solution direction as simple as possible. Furthermore, we made many small iterations to ensure substantial feedback from the industrial experts.

Our initial solution direction is based on a simple fact, a rule of thumb of our industrial experts, and two assumptions. Fact: products are composed out of many parts. Rule of thumb: the more parts products share, the more similar they are. Assumption One: We assume that the similarity measure is relative, since we think that the similarity between two products containing 1000 and sharing 900 parts is equal to the similarity between two products containing 10000 and sharing 9000 parts. Assumption Two: We assume that the similarity measure is symmetric, i.e., the similarity between *i* and *j* is equal to the similarity between *j* and *i*.

Table 1 - *Example of a product-part matrix. A product-part matrix shows which parts are contained in which products.*

product / part	$prod_1$	$prod_2$	$prod_3$	$prod_4$	…
$part_1$	1	1	1	0	…
$part_2$	1	0	0	1	…
$part_3$	0	1	0	1	…
$part_4$	0	0	1	0	…
…	…	…	…	…	…

A product-part matrix, as shown in Table 1, captures the composition of products out of parts. When a product contains a part, the corresponding value in the matrix is equal to one. In the other case, the value is zero. A product is thus represented in a column as a binary vector. As our first guess of a similarity measure between products, we took the cosine of the angle between their vectors:

$$\begin{aligned}\cos(\theta_{ij}) &= prod_i \cdot prod_j \Big/ |prod_i||prod_j| \\ &= \sum_p pp_{pi}\, pp_{pj} \Big/ \sqrt{\sum_p pp_{pi}^2}\sqrt{\sum_p pp_{pj}^2}\end{aligned}$$

where *pp* denotes the product-part matrix. Note that this similarity measure is also known as Ochiai [5].

Ochiai %	Model A	Model B	Model C	Model D	Model E	Model F	Model G	Model H	Model I	Model J	Model K	Model L	Model M	Model N	Model O	Model P	Model Q	Model R	Model S	Model T	Model U	Model V	Model W	Model X	Model Y	Model Z
Model A	100	40	39	35	33	35	36	34	34	26	26	27	27	31	29	27	36	35	38	36	35	34	33	34	34	46
Model B	40	100	92	75	71	62	64	59	59	46	45	42	41	39	36	35	70	70	73	59	57	57	53	55	54	51
Model C	39	92	100	81	78	66	68	63	64	47	48	44	45	41	36	39	72	68	72	60	58	59	58	59	60	50
Model D	35	75	81	100	93	84	87	81	81	64	64	62	62	54	50	51	61	60	63	69	66	67	69	70	70	46
Model E	33	71	78	93	100	84	80	75	76	59	61	57	58	50	46	48	62	60	60	72	75	76	76	77	77	44
Model F	35	62	66	84	84	100	96	91	91	75	77	73	74	54	53	55	54	53	56	68	67	68	71	72	72	45
Model G	36	64	68	87	80	96	100	94	94	78	78	76	75	57	55	56	54	55	58	65	63	63	67	68	68	46
Model H	34	59	63	81	75	91	94	100	99	77	77	80	79	61	60	58	52	54	57	65	64	65	67	69	69	43
Model I	34	59	64	81	76	91	94	99	100	77	78	79	80	61	59	59	53	54	56	65	64	65	66	69	69	43
Model J	26	46	47	64	59	75	78	77	77	100	99	89	89	68	72	68	36	36	37	44	43	44	50	49	49	37
Model K	26	45	48	64	61	77	78	77	78	99	100	89	90	68	70	70	38	36	37	44	43	45	50	49	50	37
Model L	27	42	44	62	57	73	76	80	79	89	89	100	99	78	77	75	36	38	39	50	50	51	53	56	56	39
Model M	27	41	45	62	58	74	75	79	80	89	90	99	100	78	77	76	37	37	38	50	50	52	53	55	56	39
Model N	31	39	41	54	50	54	57	61	61	68	68	78	78	100	96	93	45	47	48	65	63	64	69	71	70	46
Model O	29	36	36	50	46	53	55	60	59	72	70	77	77	96	100	95	40	42	43	61	59	60	68	67	67	39
Model P	27	35	39	51	48	55	56	58	59	68	70	75	76	93	95	100	43	43	44	62	60	62	69	68	69	41
Model Q	36	70	72	61	62	54	54	52	53	36	38	36	37	45	40	43	100	96	92	73	73	73	67	69	70	52
Model R	35	70	68	60	60	53	55	54	54	36	36	38	37	47	42	43	96	100	96	73	73	72	69	70	70	52
Model S	38	73	72	63	60	56	58	57	56	37	37	39	38	48	43	44	92	96	100	76	74	73	69	71	70	54
Model T	36	59	60	69	72	68	65	65	65	44	44	50	50	65	61	62	73	73	76	100	97	96	90	91	90	51
Model U	35	57	58	66	75	67	63	64	64	43	43	50	50	63	59	60	73	73	74	97	100	99	92	94	94	49
Model V	34	57	59	67	76	68	63	65	65	44	45	51	52	64	60	62	73	72	73	96	99	100	93	94	95	49
Model W	33	53	58	69	76	71	67	67	66	50	50	53	53	69	68	69	67	69	69	90	92	93	100	98	98	44
Model X	34	55	59	70	77	72	68	69	69	49	49	56	55	71	67	68	69	70	71	91	94	94	98	100	99	46
Model Y	34	54	60	70	77	72	68	69	69	49	50	56	56	70	67	69	70	70	70	90	94	95	98	99	100	46
Model Z	46	51	50	46	44	45	46	43	43	37	37	39	39	46	39	41	52	52	54	51	49	49	44	46	46	100

Figure 2 - *The similarity between MRI scanner models as measured by the Ochiai percentage. Cells are colour-coded based on their Ochiai percentage.*

Ochiai %	Model A	Model B	Model C	Model D	Model E	Model F	Model G	Model H	Model I	Model J	Model K	Model L	Model M	Model N	Model O	Model P	Model Q	Model R	Model S	Model T	Model U	Model V	Model W	Model X	Model Y	Model Z
Model A	100	13	13	12	12	13	13	13	13	5	5	9	9	11	6	5	13	13	13	13	13	13	9	13	13	15
Model B	13	100	88	75	75	53	54	47	47	33	33	33	33	26	15	20	58	61	61	50	50	49	42	46	45	32
Model C	13	88	100	84	86	59	58	51	53	37	39	37	39	30	18	27	59	53	53	48	48	50	48	52	54	30
Model D	12	75	84	100	96	71	72	63	62	51	50	49	49	36	25	30	52	49	49	55	55	54	55	59	58	29
Model E	12	75	86	96	100	71	69	61	63	49	51	48	50	35	24	32	53	48	48	56	56	57	53	57	58	31
Model F	13	53	59	71	71	100	99	90	92	79	81	76	78	37	33	41	42	40	40	45	45	47	48	52	54	30
Model G	13	54	58	72	69	99	100	91	90	80	79	77	76	37	34	38	40	41	41	46	46	45	49	53	52	30
Model H	13	47	51	63	61	90	91	100	99	80	78	82	81	46	42	39	35	39	39	44	44	43	47	51	51	24
Model I	13	47	53	62	63	92	90	99	100	78	80	81	83	45	42	42	38	38	38	43	43	46	46	51	53	24
Model J	5	33	37	51	49	79	80	80	78	100	98	89	88	42	46	43	22	24	24	25	25	25	35	34	34	15
Model K	5	33	39	50	51	81	79	78	80	98	100	88	89	41	45	46	25	24	24	25	25	28	34	33	36	15
Model L	9	33	37	49	48	76	77	82	81	89	88	100	99	60	58	54	26	31	31	35	35	35	38	43	42	21
Model M	9	33	39	49	50	78	76	81	83	88	89	99	100	59	57	56	29	30	30	35	35	37	38	42	44	20
Model N	11	26	30	36	35	37	37	46	45	42	41	60	59	100	91	85	42	46	46	63	63	62	70	75	74	29
Model O	6	15	18	25	24	33	34	42	42	46	45	58	57	91	100	93	31	36	36	54	54	53	69	67	66	13
Model P	5	20	27	30	32	41	38	39	42	43	46	54	56	85	93	100	39	41	41	57	57	60	72	70	72	21
Model Q	13	58	59	52	53	42	40	35	38	22	25	26	29	42	31	39	100	92	92	76	76	78	65	69	70	34
Model R	13	61	53	49	48	40	41	39	38	24	24	31	30	46	36	41	92	100	100	75	75	74	67	70	69	36
Model S	13	61	53	49	48	40	41	39	38	24	24	31	30	46	36	41	92	100	100	75	75	74	67	70	69	36
Model T	13	50	48	55	56	45	46	44	43	25	25	35	35	63	54	57	76	75	75	100	100	99	85	89	87	38
Model U	13	50	48	55	56	45	46	44	43	25	25	35	35	63	54	57	76	75	75	100	100	99	85	89	87	38
Model V	13	49	50	54	57	47	45	43	46	25	28	35	37	62	53	60	78	74	74	99	99	100	84	87	89	37
Model W	9	42	48	55	53	48	49	47	46	35	34	38	38	70	69	72	65	67	67	85	85	84	100	97	96	27
Model X	13	46	52	59	57	52	53	51	51	34	33	43	42	75	67	70	69	70	70	89	89	87	97	100	99	33
Model Y	13	45	54	58	58	54	52	51	53	34	36	42	44	74	66	72	70	69	69	87	87	89	96	99	100	33
Model Z	15	32	30	29	31	30	30	24	24	15	15	21	20	29	13	21	34	36	36	38	38	37	27	33	33	100

Figure 3 - *The similarity between MRI scanner models as measured by the weighted Ochiai percentage, when only three subsystems are considered relevant.*

5.1 Similarity of models

To focus on our solution direction, we took a relative small database that describes the parts that are used in models. A model is a group of products which share the same externally visible parts. A model is labelled with a unique sales name. The concept of model is widespread in industry. For example, VW Golf, VW Bora, VW Beetle, and Skoda Octavia are models in the same product family [15]. Of course, not all VW Golfs are identical, but they, at least, look the same. The similarity between MRI scanner models as measured by the Ochiai percentage[2] is visualised in Figure 2.

The experts were asked for their opinion on the similarity values. According to the experts, the similarity values were reasonable on average. The cases in which experts disagreed with the similarity values were further investigated. We learned two facts:

1) Experts do not consider all parts equally relevant. For example, the cover of an MRI scanner was considered irrelevant compared to the magnet. We even learned that the relevance of parts depends on the question at hand. For example, parts that can be easily and quickly replaced are considered less relevant by the maintainer of the test products (see section 3.3.4). The experts however indicated that specifying the relevance of each part individually was not feasible due to the large number of parts in a product. Parts have many properties, such as weight, price, time to replace, and belonging to a particular subsystem. Specifying the relevance of a part based on these properties was considered feasible. Based on the insight of the experts and since the database contained the mapping from part to subsystem, we introduced a relevance value per subsystem. Hence, the similarity measures changed to:

$$\sigma_{ij} = \frac{\sum_{s} w_s \sum_{p \in s} pp_{pi}\, pp_{pj}}{\sqrt{\sum_{s} w_s \sum_{p \in s} pp_{pi}^2}\ \sqrt{\sum_{s} w_s \sum_{p \in s} pp_{pj}^2}}.$$

Note that by setting the weight of a subsystem to zero an expert expresses that this subsystem is not in the locality set [14] of the requirement/question under investigation. An example of the similarity between models based on the changed similarity measurement is depicted in Figure 3. According to the experts, the similarity values were indeed improved by this change.

2) The experts missed information about individual products, such as the number of products sold per model. In addition, they did not consider two products of the same model identical, i.e., a similarity percentage of 100%. Unfortunately, the database did not contain information about individual products. In the next section, we describe the results using another database that contains individual product information (yet lacks the mapping from part to subsystem).

[2] We used the percentage to save two characters per value, i.e., instead of 0.*xy* we only need to write *xy*.

5.2 Similarity and difference of products

%Ochiai	Achieva 1.5T: 781196															
Achieva 1.5T: 781196	.\Subset 1	.\Subset 2	.\Subset 3	.\Subset 4	.\Subset 5	.\Subset 6	.\Subset 7	.\Subset 8	.\Subset 9	.\Subset 10	.\Subset 11	.\Subset 12	.\Subset 13	.\Subset 14	.\Subset 15	.\Subset 16
.\Subset 1	68	69	62	61	55	50	42	40	40	35	33	31	32	27	25	19
.\Subset 2	69	72	66	64	59	53	42	39	39	33	30	27	28	24	23	18
.\Subset 3	62	66	63	62	59	53	44	41	41	35	31	28	28	24	23	19
.\Subset 4	61	64	62	63	62	56	48	45	45	38	34	30	30	26	24	19
.\Subset 5	55	59	59	62	65	58	48	46	45	38	33	30	29	25	24	19
.\Subset 6	50	53	53	56	58	54	52	50	50	43	37	33	32	28	26	22
.\Subset 7	42	42	44	48	48	52	62	63	64	55	46	41	40	35	32	26
.\Subset 8	40	39	41	45	46	50	63	66	67	60	52	46	46	38	34	27
.\Subset 9	40	39	41	45	45	50	64	67	67	60	51	45	44	37	34	27
.\Subset 10	35	33	35	38	38	43	55	60	60	59	58	52	51	42	38	32
.\Subset 11	33	30	31	34	33	37	46	52	51	58	63	59	58	46	41	35
.\Subset 12	31	27	28	30	30	33	41	46	45	52	59	59	61	51	47	39
.\Subset 13	32	28	28	30	29	32	40	46	44	51	58	61	62	54	49	41
.\Subset 14	27	24	24	26	25	28	35	38	37	42	46	51	54	56	56	50
.\Subset 15	25	23	23	24	24	26	32	34	34	38	41	47	49	56	60	57
.\Subset 16	19	18	19	19	19	22	26	27	27	32	35	39	41	50	57	62

Figure 4 - *The average similarity as measured by the Ochiai percentage between sets of MRI scanners of a single model. The sets of MRI scanners are created by ordering and grouping the MRI scanners based on their ordering date. The matrix of similarity values shows the evolution of a single model over time.*

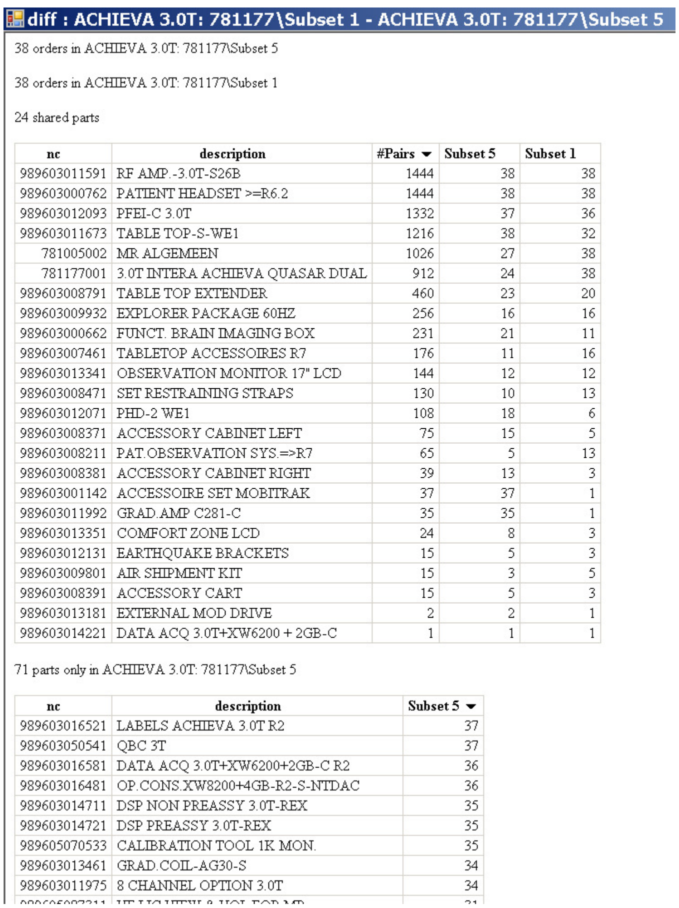
Whereas the number of models is relatively small, the number of products sold is considerably larger. As a consequence, a similarity matrix containing all products is too large to fit on a single page or screen. For this reason an (artificial) hierarchy in products had to be added. At the highest level, a division in models seems logical. Since the number of products sold of a single model is still too large to fit on a single page, we had to order the products even further. A number of options exist. Subsets can, for example, be made arbitrarily, by clustering based on similarity, and by clustering based on date. We opted for clustering based on date, since it would enable the visualization of the evolution of models in a product family over time. Still, a number of dates are associated with a product: order date, production date, and delivery date. Ordering products based on these dates adds some structure, since in general the closer the dates, the more similar the products, but exceptions exist in all three cases:

- Order date: while most customers want their products as quickly as possible, in some cases the hospital still had to be built when an order for an MRI scanner was placed.
- Production date: the production date of a product is ill defined. Not only since part of the integration happens at the customer's site, but also since the production dates of the constituent parts vary considerable.
- Delivery date: a product can be produced considerably earlier than delivered, for example, due to variations in

available capacity in the factory caused by the variation in orders and availability of personnel over time.

Since all kinds of dates had their drawbacks, we just selected one: the order date. The experts were asked for their opinion on the similarity values calculated based on a large database originating from logistics. The similarity values were presented using the previously described hierarchical structure. See also Figure 4.

The experts were surprised by the huge difference between the initial product and the current product of the same model. For example, in Figure 4, we see that the average similarity is just 18 percent between subsets 2 and 16. As a consequence, they desired more details about this difference. We addressed this desire by listing the shared and different parts between (groups of) products as is shown in Figure 5. Note that this information is crucial to be able to increase the similarity in the installed base (see section 3.3.2).



diff : ACHIEVA 3.0T: 781177\Subset 1 - ACHIEVA 3.0T: 781177\Subset 5

38 orders in ACHIEVA 3.0T: 781177\Subset 5

38 orders in ACHIEVA 3.0T: 781177\Subset 1

24 shared parts

nc	description	#Pairs	Subset 5	Subset 1
989603011591	RF AMP.-3.0T-S26B	1444	38	38
989603000762	PATIENT HEADSET >=R6.2	1444	38	38
989603012093	PFEI-C 3.0T	1332	37	36
989603011673	TABLE TOP-S-WE1	1216	38	32
781005002	MR ALGEMEEN	1026	27	38
781177001	3.0T INTERA ACHIEVA QUASAR DUAL	912	24	38
989603008791	TABLE TOP EXTENDER	460	23	20
989603009932	EXPLORER PACKAGE 60HZ	256	16	16
989603000662	FUNCT. BRAIN IMAGING BOX	231	21	11
989603007461	TABLETOP ACCESSOIRES R7	176	11	16
989603013341	OBSERVATION MONITOR 17" LCD	144	12	12
989603008471	SET RESTRAINING STRAPS	130	10	13
989603012071	PHD-2 WE1	108	18	6
989603008371	ACCESSORY CABINET LEFT	75	15	5
989603008211	PAT.OBSERVATION SYS.=>R7	65	5	13
989603008381	ACCESSORY CABINET RIGHT	39	13	3
989603001142	ACCESSOIRE SET MOBITRAK	37	37	1
989603011992	GRAD.AMP C281-C	35	35	1
989603013351	COMFORT ZONE LCD	24	8	3
989603012131	EARTHQUAKE BRACKETS	15	5	3
989603009801	AIR SHIPMENT KIT	15	3	5
989603008391	ACCESSORY CART	15	5	3
989603013181	EXTERNAL MOD DRIVE	2	2	1
989603014221	DATA ACQ 3.0T+XW6200 + 2GB-C	1	1	1

71 parts only in ACHIEVA 3.0T: 781177\Subset 5

nc	description	Subset 5
989603016521	LABELS ACHIEVA 3.0T R2	37
989603050541	QBC 3T	37
989603016581	DATA ACQ 3.0T+XW6200+2GB-C R2	36
989603016481	OP.CONS.XW8200+4GB-R2-S-NTDAC	36
989603014711	DSP NON PREASSY 3.0T-REX	35
989603014721	DSP PREASSY 3.0T-REX	35
989605070533	CALIBRATION TOOL 1K MON.	35
989603013461	GRAD.COIL-AG30-S	34
989603011975	8 CHANNEL OPTION 3.0T	34

Figure 5 - *Shared and (part of) different parts contained in two different sets of a single model.*

Showing the actual parts in products triggered an interesting discussion with the experts: What is a part? The database was maintained by logistics and made distinctions between parts that were painted in different colors and between pieces of hardware with minor change, e.g., due to bug fixes and cost reductions. For some experts, these small differences made the parts indeed different. For others, these small differences were irrelevant, and they would like to consider these parts as being identical. Fortunately, the parts were encoded such that the least significant bits encoded these small differences, such that we were able to use the appropriate definition of a part for each expert.

Another point of feedback we received from the experts was related to the products shown. For many questions not all products are relevant. For example, to answer the question which products to test for a replacement (see section 3.3.1) only those products that use the part to be replaced are relevant. An expert thus wants to be able to filter the product presented to suit his needs.

6 Summary

To support evolving product families, being able to measure the (dis)similarity between products is a prerequisite. In an industrial setting, we experimented with a similarity measurement and compared the resulting similarity values with the opinions of experts.

We learned that the measurement depends strongly on the questions at hand. Hence, a similarity measure must be highly configurable, including

- Specifying the relevance of parts, e.g., using a weight per subsystem.
- Specifying the definition of a part, e.g., can a part have multiple coloured instances?

Furthermore, the presented information must be highly configurable, including

- Ordering the products in a hierarchical structure to enable visualization.
- Filtering the products to exclude irrelevant products, e.g., only present products using a given part.
- Focus on the similarity and difference between (groups of) products to get more insight in the causes of a similarity value.

Finally, our similarity measurement turned out to be in good agreement with the experts' opinions. We think similarity measures are valuable for industry. Illustrative of this industrial value for us was the fact that one of our earliest results had a prominent place on the wall of one of the experts for a couple of months.

7 Discussion

We kept our solution direction as simple as possible. Consequently, we just counted parts. However, many problems are caused by interactions between the constituent parts of a product [6]. Since testing combinations of parts achieves better results only in particular cases [2], we doubt whether counting combinations is better in general. Yet, whenever evidence becomes available that counting combinations of parts, i.e., pairs, triplets, quadruples, etc., would yield better results than counting individual parts, we would definitely go for these more complex solutions.

While developing a similarity measure, we heavily relied on experts. Experts are currently judging the similarity between products based on experience and gut feeling. Discussions with the experts revealed that their accuracy is at most in the order of tens of percents. Our solution direction, in which parts are counted to calculate the similarity values, seems to yield exact numbers. We, however, consider the accuracy of our similarity values comparable to that of the experts, i.e., tens of percents, among others, because experts were used in the evaluation of the similarity values; because configuring the similarity values needs expert input; and because the databases were not designed for extracting exact similarity values.

8 Road ahead

We would like to investigate whether the transfer function of knowledge about product *i* to product *j* is equal to the similarity value between product *i* and *j*. For example, if all tests succeed on product *i*, is the chance that all tests succeed on product *j* given by the similarity value between products *i* and *j*? Furthermore, we would like to generalize this question: what is the transfer function of knowledge about a set of products to another product? Is it the maximum of similarity values between a product in the set and the other product? Does it relate to the total amount of parts shared? Or, are pairs, triplets, or even quadruples needed [6]? By answering these questions, we are closer in answering the industrial questions of sections 3.3.1, 3.3.3, 3.3.4, and 3.3.5.

In product families, one typically has a minimal product, i.e., without any optional functionality, and a full-fledged product, i.e., with all optional functionality. We have the gut feeling that the similarity between the minimal and the full-fledged product is not symmetric. For example, we think that the chance of all tests succeeding on the minimal product given that all tests succeeded on the full-fledged product is higher than the chance of all tests succeeding on the full-fledged product given that all tests succeeded on the minimal product. Focusing on minimal and full-fledged products, we would like to challenge our second assumption: the similarity measure is symmetric, i.e., the similarity between *i* and *j* is equal to the similarity between *j* and *i*.

Within Philips Healthcare MRI, we expect that showing the valuable information for the development and support of evolving product families stored in the different databases will result in aligning the existing databases and in changes in the information contained in them to make it usable throughout the whole organization instead of a single department, such as logistics. When the content of the databases is collected with the requirements of similarity measures in mind, the accuracy of the similarity values will definitely increase.

When looking into the future, we expect industry to be confronted more and more with problems related to supporting evolving product families. Hence, we expect that supporting evolving product families will receive the attention it deserves soon, not only from industry but also from the academic world. This article was written with the goal of drawing attention to the challenging problems associated with supporting evolving product families. We hope we did achieve this goal!

9 Acknowledgements

We would like to thank the experts of Philips Healthcare MRI for their useful feedback. In particular, we thank Jeroen de Jong for his enthusiasm and support. Furthermore, I would like to thank Gerrit Muller and Pierre America for useful feedback on an earlier version of this paper.

10 References

[1] Al Dallal, J. & Sorenson, P. (2008), “Testing Software Assets of Framework-Based Product Families during Application Engineering Stage”, *Journal of Software* 3(5), 11- 25.

[2] Bach, J. & Schroeder, P. (2004), “Pairwise Testing: A Best Practice That Isn't”, *Proceedings of 22nd Pacific Northwest Software Quality Conference*, pp. 180-196.

[3] Bertolino, A. & Gnesi, S. (2003), “PLUTO: A Test Methodology for Product Families”, *Proceedings of the Fifth Workshop on Software Product Families Engineering*, pp. 181-197.

[4] Eriksson, M. (2003) “An Introduction to Software Product Line Development”, *Proceedings of Umeå's Seventh Student Conference in Computing Science*, pp. 26-37.

[5] Esposito, F., Malerba, D., Tamma, V. & Bock, H.-H. (2000), “Classical Resemblance Measures”, in Bock, H.-H. & Diday, E. (eds), *Analysis of Symbolic Data: Exploratory Methods for Extracting Statistical Information from Complex Data*, pp. 139-152.

[6] Grindal, M., Offutt, J. & Andler, S.F. (2005), “Combination testing strategies: a survey”, *Software Testing, Verification and Reliability* 15(3), 167-199.

[7] Herald, T.E., Verma, D. & Lecher, T. (2007), “A Model Proposal to Forecast System Baseline Evolution due to Obsolescence through System Operation”, *Proceedings of fifth annual Conference on Systems Engineering Research*.

[8] McGregor, J.D. (2001) *Testing a Software Product Line*, Technical Report CMU/SEI-2001-TR-022.

[9] Muccini, H. & van der Hoek, A. (2003), “Towards Testing Product Line Architectures”, *Electronic Notes in Theoretical Computer Science* 82(6), 99-109.

[10] Muller, G. (2008) *Modeling and Analysis: Life Cycle Models*, http://www.gaudisite.nl/MAlifeCyclePaper.pdf

[11] Nebut, C., Fleurey, F., Le Traon, Y. & Jézéquel, J.-M. (2003), “A Requirement-Based Approach to Test Product

Families", *Proceedings of the Fifth Workshop Product Families Engineering*, pp. 198-210.

[12] Rowe, D., Leaney, J. & Lowe, D. (1998), "Defining systems evolvability–a taxonomy of change", *Proceedings of the International Conference and Workshop on Engineering of Computer-Based Systems (ECBS '98)*, Jerusalem, Israel, pp. 45-52.

[13] Sandborn, P.A., Herald, T.E., Houston, J. & Singh, P. (2003), "Optimum Technology Insertion Into Systems Based on the Assessment of Viability", *IEEE Transactions on components and packaging technologies* 26(4), 734-738.

[14] Scheidemann, K.D. (2006), "Optimizing the Selection of Representative Configurations in Verification of Evolving Product Lines of Distributed Embedded Systems", *Proceedings of the 10th International Software Product Line Conference (SPLC'06)*, pp. 75-84.

[15] Suh, E. (2005), "Flexible Product Platforms", PhD Thesis, Massachusetts Institute of Technology (MIT), USA.

[16] Tevanlinna, A., Taina, J. & Kauppinen, R. (2004), "Product Family Testing – a Survey", *ACM SIGSOFT Software Engineering Notes* 29(2).

[17] Tseng, M.M. & Jiao, J. (1998), "Design for Mass Customization by Developing Product Family Architecture", *Proceedings of Design Engineering Technical Conferences (DECT)*.

[18] van der Linden, F. (2002), "Software Product Families in Europe: The Esaps & Café Projects", *IEEE Software* 19(4), 41-49.

[19] van de Laar, P., America, P., Rutgers, J., van Loo, S., Muller, G., Punter, T. & Watts, D. "The Darwin Project: Evolvability of Software-Intensive Systems", *Proceedings of Third International IEEE Workshop on Software Evolvability*, pp. 48-53.

[20] van Ommering, R. (2004), "Building Product Populations with Software Components", PhD Thesis, University of Groningen, The Netherlands.