People Management in Institutionalizing Product Lines

Klaus Schmid

Fraunhofer Einrichtung für Experimentelles Software Engineering,
Sauerwiesen 6, D-67661 Kaiserslautern, Germany
+49 (0) 6301 707 – 158
schmid@iese.fhg.de

Abstract.[1] Currently, the key approach to large-scale software reuse is product line development, an approach built upon the systematic development of building blocks for reuse. While most work on product lines focusses on technical issues, in the end they are built by by people. Winning them for product line development is a key issue in successfully introducing and institutionalizing product line development in an organization. This makes people issues a key concern in technology transfer. Based on our experience in technical transfer in the product line area we developed some hypotheses on how these people issues can be formed in a way favorable to product line development. In particular, we focus on the relation between the *mind set of people*, *communication patterns*, and the *organizational structure*. We then give concrete rules based on our own industrial experience as well as industry reports that support instutionalizing product line development.

Keywords. Domain Engineering and Software Product Lines, Business Issues, Architecture-Centric Development, Organizational Structure, Communication Patterns, Technology Transfe

1 Introduction

Product line development has surfaced over the last decade as a successful approach to large-scale reuse [1, 2]. This approach relies on the explicit identification of a number of systems (a product line) that should be targeted by the reuse initiative. Based on this, a domain engineering is performed, aiming the development of assets that are reusable across the range of the product line. These assets are then reused in application engineering. Application engineering is responsible for targeting the specific system under development, selecting and assembling the reusable parts (according to the product line architecture) and implementing parts which are specific to the final systems. Transitioning an existing, one-system-at-a-time development organization to product line development has proven to be a rather demanding task.

---

[1] This work was supported in part by Eureka Σ! 2023 Programme, ITEA project ip00004, Café.

In this paper we aim at a contribution towards answering the question: *how can we successfully support organizations in their shift towards product line development*?

In current research on product line development, the focus is often set on technical issues: what is the best way to do scoping, how can we best capture results of domain analysis, how can we support the description of variability in reference architectures, etc. [14, 15]. While these are very important questions to ask (and to answer), they leave out what we regard as a major aspect of technology transfer in the context of product line development: the importance of people..

Thus, in this paper, we want to focus exclusively on people issues, i.e., aspects like the *believes of people*, the *social situation*, they are acting in, or their *relationship to the overall software development organization*. From our work in transferring software product line approaches into industrial practice we often saw how important people issues are [8, 11]. This is especially true, if we focus on transforming a single system company into a product line organization.

Driven from the motivation of technology transfer, we focus on the question of what does it mean to get the people issues right and what are its implications on technology transfer. As opposed to other key work in this area we will focus on the perspective of the individual developer [4, 18].

From an empirical point of view, this work relies foremost on our personal experience in product line development in industrial practice. Over the course of the last seven years, we had the opportunity of working together with a number of companies ranging from small to large on the topic of product line development and accompanying their product line initiatives. This direct experience was augmented with many discussions and, of course, a systematic analysis of existing, documented case studies of product line development.

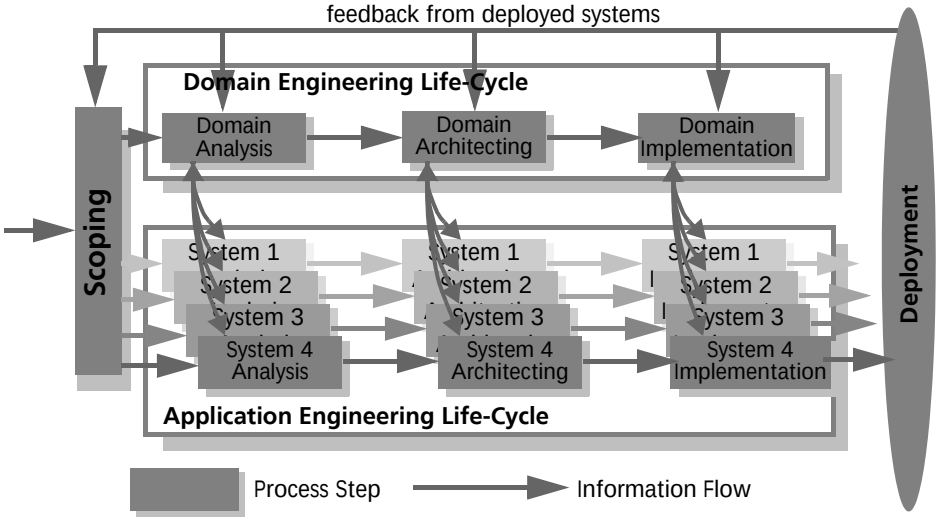


Fig. 1. The Two-Life-Cycle Model of Software Reuse

In Section 2 we will give a more detailed description of the issues on which we want to focus here and the underlying concepts we are going to use. In Section 3, we will then discuss the key aspects of our model and what we learned from our experiences as well as from other reported experiences on technology transfer in product line development. In Section 4, we will summarize our main hypotheses as success factors for product line technology transfer. Finally, in Section 5 we will conclude.

2 Scope of the Analysis

Here, we treat people issues in a rather broad manner, including communication aspects and organizational concepts. People issues, as we define it, denotes all aspects relating to the individual person participating in the software development process and its relationship to the overall software development organization, including also the team and the organizational structure.

Basically, every experience report (e.g., [5, 6, 7, 11]) or discussion on technology transfer issues (e.g., [3, 11]) addresses people issues as well. What is missing from our point of view is a description of how these aspects interact into *providing a supportive environment to product line development* and comparing this with experience from the field. One of the goals of this paper is to provide a model of this interaction. Based on our experience in technology transfer in product line development we found:

The dominating success factor for product line development is the mind set of people, i.e., the interpretations people give their experiences as well as their personal motivations and goals.

With this we mean, that only if we are successful in making reuse and the integrity of the product line an important goal for the individual developer, we will be successful in making product line development a reality in the corresponding organization. Luckily, this does not mean that we are at the mercy of what is on the mind of people in the development organization, but we believe that this mind set can be actively influenced. Actually, we are certain that it must be changed on the way from a single system organization to a product line organization in order to make technology transfer successful.

Here, we will focus on two main aspects: *communication patterns* – what is communicated to whom – and *organizational structure* – which stable role assignments are given to which person – and what dependencies exist among the people. From our point of view dependencies exist among these aspects as is illustrated in Figure 2. However, this picture is only meant to be illustrative, not a complete description of all relationships. At all levels different, additional influences that are not depicted come in. For example, we do not make the assumption that the mind set is *only* influenced by the communication patterns. We will actually discuss further influences in this paper. The picture is meant to highlight the three main areas (mind set, communication patterns, and organizational structure) which we will discuss in this paper along with the relation among them. In addition, it is important to note that this figure is no static view, rather it implies that if you change the higher levels, like the organizational structure, the lower levels, we are actually aiming at, will follow.

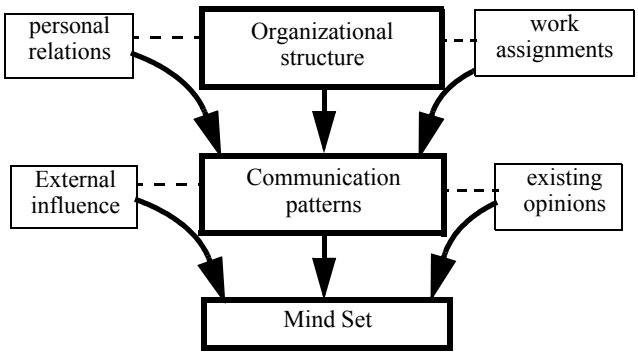


Fig. 2. Schema of influences of soft factors

In this paper, we use the term *mind set* to denote any aspect like *belief, attitude, priority,*or *value*, that is a driving force in determining the behavior of developers. We will focus on the specific priorities and believes people have in mind, which in turn determines what they focus upon in their work and where they invest their time in.

The assumption expressed by Figure 2 is that the mind-set is influenced by soft factors, in particular, the *communication* people are involved in.[2] In any work-place environment these communications are structured in a certain way. Specific types of information are usually communicated with specific people depending on their needs and interests. If these relations are somewhat stable over time and not people-specific we call them *communication patterns*. A typical communication pattern in non-product line organizations, which can be often observed in practice, is that developers mourn to each other that they have to sacrifice the integrity of the architecture of the software system for making a deadline.

The communication patterns are partially determined by the overall work organization, as they influence what information needs to be communicated to which person. The *organizational structure* determines who is concerned with which part (functionality-wise) and which aspect (life-cycle perspective) of the product line. Again, this influence is not a complete one. Rather, the organizational structure provides constraints on the overall communication patterns. Thus, the organizational structure also has in an indirect manner a strong impact on the mind set of people.

The fact that the organization structure, the communication patterns, and the resulting products a company develops are strongly interrelated has already been emphasized by the classical quote by Conway [17]:

> **Organizations which design systems are constrained to produce systems which are copies of the communication structures of these organizations**

As a corollary we can argue that an organization that focuses on product line development needs to have communication structures that mirror the product line. We will discuss in Section 3.2 the consequences of this.

---

[2] It is also influenced by previous experiences, the raising of the people, etc. Aspects that are well studied in psychology, but beyond the scope of this paper.

Here, we advocate that this relationship can be extended to product line engineering: our core problem is to find adequate organizational forms and communication structures that mirror the needs of product lines.

3 Developing a product line perspective

In the previous section we discussed how mind set, communication patterns, and organizational structures interrelate. In this section, we discuss in more detail how they interact in a product line context and how they should be formed in order to support the introduction of product line development.

3.1 Mind Set

When discussing the mind set of software developers in the context of this paper, we will focus on the believes and priorities they have with respect to their working goals. This determines the intrinsic motivation people have to focus on and care for successful product line development. As product line development today strongly depends on the personal qualification and motivation of people (even more than other forms of software development), we believe this to be the very key aspect that drives successful product line development.

In terms of a positive mind set for product line development we would obviously expect people to show the following believes and priorities:

- They believe in the product line concept, in the sense that they believe to be possible to design systems in such a way that major parts can be reused across the development of multiple systems.
- They believe it to be necessary and possible to do good engineering work, in the sense that they want to achieve high-quality and highly efficient solutions.
- They feel responsible for the well-being of the whole, i.e., not simply for the specific sub-task they are assigned to. In particular, they accept to take their share in ensuring the integrity of the product line.

In a successful product line organization it is necessary that people have these priorities and believes, as those drive their actions. In particular, developers in a product line organization should show the following traits in their behavior:

- They need to develop software so as to be reusable, i.e., even down to the very details of coding the right decisions for reuse are made.
- We expect them to adhere to architectural principles. It is common wisdom that the integrity of the product line architecture is key in ensuring the long term success of a product line. However, defining a product line architecture is one thing, ensuring that each and every developer adheres to these principles is a different thing. This is the easiest, if each and every developer believes in the necessity of this.
- Finally, we expect them to actively search for and explore reuse opportunities (really use what has been provided and not reinvent the wheel, and also to identify reuse potential on top of the one identified in architecture development).

The soft factor of appropriate mind set is one of the key issues in this paper. We found it to be key in all software development organizations which successfully perform product line development (e.g., [10]).

For example, we cooperate with a medium-sized company which actively "invented" major aspects of product line development out of their need to develop variants of their products and the presence of this mind set. On the other hand we know of a similarly sized organization which explicitly aimed at product line development, but had only limited success as people always focused on the differences among the products without seeing the commonalities, thus the necessary mind set was not present. We even know of rather big companies that actually have a product line organization in principle in place, but have only limited success in product line development as the required mind set is still missing.

From our experience the mere believe that commonalities exist, are worthwhile to make explicit, and can actually be exploited strongly changes the overall behavior of people and thus the success of product line development.

In summary, we expect in a product line organization people to show this behavior:

1. They aim at avoiding rework, i.e., they work pro-actively towards making it easier to integrate future products. In order to enable this we need to make the product line visible to the people and even more to make it a personal experience to people that they can reduce the amount of effort they need by having this attitude.[3]
2. Related to this is that people see the overall product line infrastructure and actively work towards restructuring it as an high priority task, if they see problems for the product line potential.
3. People should focus on the product line instead of the product, i.e., the predominant question for them should be: what does the change I am about to make mean for the whole set of products our organization is developing?
4. The people should value the product line higher than the individual product. An obvious sign for this is the willingness to sacrifice a product deadline over the conceptual integrity of the product line. This is of course something that needs to be supported and communicated by management, as we will discuss below.

All these attitudes taken together form a common principle:

Developers need to feel responsible for the well-being of the product line as a whole, as opposed to the traditional single product focus.

As we see it, the mind set is mostly determined and reinforced by the communication patterns, which we will discuss in the next section. In addition, the mind set is obviously influenced by many external aspects like the overall background, previous and current experience outside the work-place, and so on. However, most of these issues are beyond our control in a product line context. Thus, we focus here on the *one* we can influence: communication patterns.

---

[3] Note, that we emphasize rework instead of reuse here for two reasons: first we regard reuse as a means to an end, namely saving effort; second this was the attitude that we experienced in successful product line organizations.

3.2 Communication Patterns

With communication patterns we mean here typical information flows among people, viewed from a content perspective (instead of a structural perspective), including complex interaction patterns. The potential communication patterns are obviously dependent on the specific interactions among people that can happen. These are in turn dependent on the overall organization, which we will discuss further in Section 3.3.

As the mind set can not be influenced directly, it is important to take other angles. From our point of view the most important possibility is to influence it – to the extent possible – through communication. This holds both for a technology transfer situation, where people must be acquainted with these attitudes in a step-wise manner, as well as during the continued development of a product line, where these attitudes must be supported and kept up.

Thus, these attitudes must be continuously communicated to the product line development personnel. The following aspects are particularly important to communicate:

- People must feel involved and must be convinced.
- People should experience personal benefits from product line engineering.
- The product line must be made visible to people.
- The potential for reusing across this product line must be made visible.
- The need for applying good engineering practices must be communicated.

The following means can help to communicate this. (Here, we focus on their communication aspect):

1. Trainings
2. Champions
3. Management messages
4. Involving practice

Each of these communication forms has its unique set of advantages and disadvantages. Out of them *trainings* are probably the most straightforward. In the context of technology transfer they are typically used when people are trained for new technical activities that are particularly relevant for product line development. Typical of this are trainings on domain analysis or architecture concepts. This can and obviously should be used to reinforce the need for product line development and to understand any problem people have with this shift in attitude, thus making it a two-way communication. The advantage of this communication form is that the time of people is explicitly reserved for this activity allowing them to focus on it. Also in such a forum any difficulties and problems can be addressed more directly than in everydaily practice. The disadvantage of trainings is that this is often a one-time activity, which is on a rather abstract level for the developers, as it predates their practical involvement. It will only show results, if the communicated information and attitudes are found practical in everyday experience (cf. *involving practice*). We also found that continuous coaching over a prolonged period of time is key in turning training results into standardbehavior.

The results of trainings can easily be negated in daily practice. Often this effect happens through management messages. Typical cases are that only insufficient funds for trainings and subsequent coachings are available, the management continues to value the individual project deadline higher than the product line integrity. In all these cases easily frustration is created for the people with obviously non-beneficial results.

Different interpretations of the term "*champion*" are used in literature. Here, we mean a person with strong social and communication skills who is in continuous contact with the development personal and is in favor of product line development ideas and communicates this strongly and convincingly. The champion may have, but need not have a management role (cf. *management messages*). The champion helps to keep up commitment for product line practices and helps other people in seeing the product line. There are no disadvantages of champions as far as we can see.[4] However, there is not always an appropriate person available in the environment. In the worst case it can also be that a person who would have the necessary skills to fit this role is a strong opponent of product line practice. This is about the worst thing that can happen to a product line program.

As very important for changing practices in an organization we found the *messages management* sends. With *messages* we do not mean so much the general speech management is giving but more the indirect signals it is sending. For example, suppose a manager strongly claims support to product line concepts. However, he is willing to sacrifice the product line integrity by making architectural shortcuts in order to meet a product deadline. This message will come across and will have a strong effect on the people, as they will feel evaluated more meeting the deadline then based on contributing to the product line. Similar messages are:

- *sufficient funding* – does management provide sufficient funding (both money and personnel) to support product line development.
- *a strong product line vision* – e.g., management is willing to forgo a customer contract that does not fit the product line vision.

From our experience, as well as from other experience reported in literature [3, 6, 7] we believe that management commitment is a key factor in product line success. This is also related to the organizational structure, which we will discuss in Section 3.3.

While all aspects mentioned above are contributing to a product line attitude in an organization, we believe the aspect of *involving practice* to be key. With the term *involving practice* we mean: the way of working the individual person experiences in everyday practice. We found that only if this changes – and people can relate benefits to the change – they will accept and support the change to product line practice.

Practice strongly depends on management decisions and in particular the organization of work (cf. Section 3.3). Other important aspects are the personal involvement in product line aspects and the personal benefit for people through product line thinking.

In order to make the product line visible we need to achieve a common understanding among people involved in the development of the different products. This is often a difficult task as people may have different vocabularies, different perspectives and different backgrounds. While all of these aspects are important for successful product line development, so far only the development of a common vocabulary as a core task of domain analysis has received considerable attention (e.g., [6, 9]). If a common terminology has been established, the exchange of people between product development teams is another way of making the product line visible throughout the organization. In

---

[4] However, a "champion" who is not in favor of product line engineering can be a major problem for any product line initiative.

some companies we found that the product line was insufficiently visible. This can happen for example, in case different business units are responsible for developing certain products of the product line. It can also happen if the various variants are distributed too strongly in time.

Sometimes also key developers are supposed to play the role of a champion but do not believe in product line development. In other situations we saw management to make it pretty clear that the project deadline is the ultimate priority. These attitudes always have proven to be a threat for introducing successful product line development into an organization. In every case we observed the major palyer was not convinced of product line engineering, the effort failed.

Similarly we also found situations where the opposite was true: key people were strongly pro product line, the current project was not valued higher than the product line, and overall a very positive attitude towards software reuse existed. In all these situations we saw a strongly different mind set with the developers and accordingly strong differences in terms of the actual software development and reuse results.

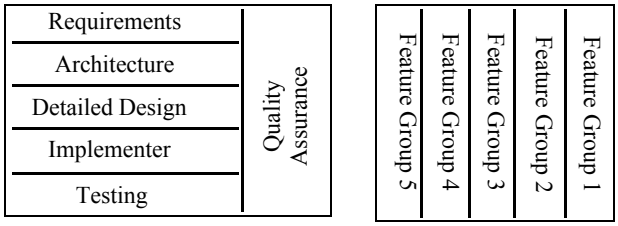
Out of the four different forms of communication patterns, we discussed in this paper, the first three can actually be manipulated more or less at will (which does not imply it is easy to do appropriately and consistently). Only the last one is more difficult. This is actually dependent on how the work is organized overall. We will discuss this in more detail in the following section.

3.3 Organizational Structures

Here, we do not restrict ourselves to the "official" organization, but we look at any form of at least temporarily stable role-relationships. This is a broader view as is taken in other work on this topic (e.g., [13]). The organization of work plays a predominant role in structuring the practice of people and what they need to communicate about. In terms of the communication patterns we described above, this impacts especially the aspect of involving practice with all the consequences connected to this. In particular, this has a strong impact on whether people will be able to see the product line. The tasks people have to work on, the problems they have to tackle, etc. are strongly influencing the perspective of people (i.e., their mind set), as they will determine whether considering product line issues are also the best solution for the individual.

Here, we want to distinguish *formal* and *informal organization*. The formal organization determines who is officially in charge of what aspect of work, who has management responsibility for which part of the organization, how are the management hierarchies structured, etc. A good overview of different formal organizational models that are used in the context of product line development is given in [4]. On the other hand we have the informal organization, relating to temporary working groups, as well as semi-official bodies. These are typically not found on the organization's structure chart, but they have a strong impact on the communication patterns in the organization (an example of this is the architectural steering board in the Owen approach [5]).

Two fundamentally different possibilities for constructing a software development organization exist: phase-wise and feature-based organization. These map roughly to the process- and product-oriented organizations as they are termed in [16]. Other organizational structures like the layered or technology-oriented structures, or prototype



a) Phase Organization b) Feature Organization

Fig. 3. Organizational Prototypes

teams, can be regarded as special cases of the feature-organization. If we look at the formal organization within a business unit, we often see a phase-based organization. However, from our experience, we came to the conclusion that in terms of an organizational structure the predominant structure should be based on features, not development phases.

Note, that these are extreme cases. Real organizations usually lie in-between. For example, the per-feature organization is typically restricted to detailed design, implementation, and unit testing (where a single person can be responsible for more than one of these phases).

In a featured-oriented organization certain development groups are responsible for certain aspects of the software in terms of functionality. This organization mode is particularly meaningful in the case that the same group is responsible for this functionality across different products. An example of this is again the Owen-organizational model (cf. [5]) where the same person is always responsible for the same code-module. We also saw this working in practice, where across a range of different products the same person was always responsible for the same functionality.

From our point of view the feature-based organization holds important promises for product line development. As the same people are charged with developing similar functionality for different products, this leads to two important effects:

- It becomes easier to ensure the integrity of the architecture as the same person responsible for a part of it also makes changes to it.
- People experience a *personal* benefit in their work by focusing on reusability aspects.

The repercussions of this organizational mode on the mind set are clear: Suddenly, it becomes the personal benefit of the developer to focus on reuse.

As the organizational modes described above are rather extreme, for large development projects a combined mode of operation is typically the most appropriate. For example, a certain group might be responsible for the overall requirements and the customer relationship, while detailed requirements, design and implementation are developed by a different group, structured according to features (main functionality areas). Integration of the software as well as end-user testing (as opposed to module level testing) is routinely made by a different group which may actually overlap with the first group. At the same time the overall architectural integrity of the product line may be ensured by designated architects or an architectural steering board composed of members of the second group.[5] An example of these principles is the Owen

approach [5] which is based on an architectural steering board composed of members of the different product teams. Another example is the organization used at CelsiusTech [7] which has only requirements personnel in the application engineering units, while development is completely done in the domain engineering unit.

The organization forms exhibited by CelsiusTech [7] and Owen [5] are actually two ends of a spectrum as is shown in Figure 4, which is based on a figure from [5]. In extreme cases, dedicated application engineering personnel can vanish completely [7]. This maps to the domain engineering unit model described in [4]. However, if we look at the two cases more closely, we see that while the large difference holds for the formal organization, in terms of the informal organization the they are actually rather close together, as in terms of the informal organization both companies mostly adhere to an organizational model based on features or components. As in both cases the focus is on the common functionality (features) and their implementations (components).

From the case studies we analyzed [5, 6, 7, 9], as well as from our own industrial experience [8, 10, 11], we came up with the following organizational rule:

The larger the reuse potential in a product line and the larger the exploitation of this potential, the stronger should feature-orientation dominate the organizational structure.

At this point it does not really matter whether the feature-orientation is anchored in the formal organization or in the informal organization, it is only important that it exists in a form that impacts the everyday work of the software developers. As a corollary, we can also say that in the case of an organizational structure with explicit domain engineering, *the domain engineering unit should be structured in a feature-oriented way.*

Besides being able to experience personally the need for reusability, an organizational structure based on the described principles is key to enable the developers to see the product line at large. This is very important for making the individual developer feel involved and responsible for the product line at large. Such an organizational structure is necessary to create the involving practice, described in Section 3.2, that is necessary to focus the developers on product line concepts. In terms of creating the appropriate mind set it is also very important in order to strongly involve people in the creation of a product line perspective.

In particular, this implies that a large number of people can contribute to what is regarded as the product line both now and in the future. This implies that the product roadmap is not only defined by management and marketing. In the context of Cummings [6], this was emphasized as a major success factor. At best, the involvement of people is direct, or at least a stakeholder of each group should be involved. Whatever approach is used, it is of key importance to involve as many stakeholders as possible [9]. If the groups are rather large, then again we can make use of a feature-based decomposition. The different people in a group of developers are made responsible for different functionality areas and then contribute accordingly to the different across-

---

[5] This highlights also a different aspects: it is a requirement that an organizational body – be it a manager or a board of people feels responsible for the product line. We are aware of cases, where assigning a manager as the head of a group of product units brought already a shift towards product line reuse.

product groups. Thus, for each product division stakeholders are integrated into the definition of each part of the product line description and everybody (or at least most people) are integrated in this process.

This broad involvement is particularly important so that across an organization a common language can be discovered. But stakeholder involvement has reportedly an important effect in the context of transition to a product line as well:

- It helps to reduce the fear of change people have.
- It makes them feel responsible for the results.
- It helps to avoid the *not-invented-here* syndrome.

Altering the organizational structure is actually something that can be done in an explicit and purposeful way. As the informal organization can usually be more easily changed than the formal one, this is usually the typical starting point to make other pieces (communication patterns and mind set) fall in place. Starting with the informal organization also has the advantage that it is much more easy to alter incrementally. We can start with changing the development mode for certain products or parts of the functionality and then extend this in a stepwise manner, affecting the working mode of only those people relevant to this functionality and products.

4 Organizational Success Factors

In this section, we want to summarize some main principles that can be extracted from the discussion given above along with some potential measures that can be used to support them. From our point of view these issues should be key concerns when introducing product line development in an organization:

From an organizational point of view one of our key principles is that the product line structure should be mapped onto the product line organization. This entails:

The same people should be responsible for the same functionality (organization follows software architecture). This implies a restricted use of a phase-wise work-breakdown and a shift to a more feature-based organizational mode.

This measure does directly relate to the communication principle of demonstrating to people the potential for reusing across the product line and it focuses on the mind set principle of making the product line work personally beneficial to people.

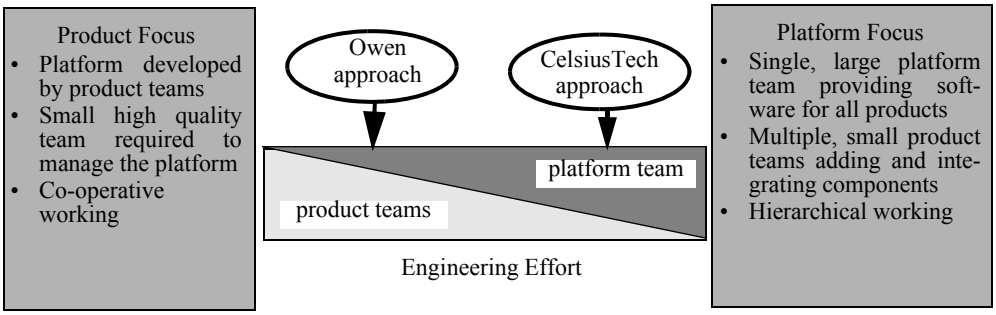


Fig. 4. Product focused vs. platform focused development organizations [5]

Another key principle we found is that the product line needs to be made visible to the relevant people. In terms of the product line organization several potential measures for ensuring this exist:

- In the case of small organizations (less than 10 people): bring the relevant people together. Make them work together on the product line vision.
- For large organizations select subsets for the different activities (e.g., developing domain analysis)
  - assign people based on topics
  - select representatives (for very large organizations) to do the modeling
- In case of multiple divisions, exchange personal

This works together to create a shared vision among people. It also helps them to see the full extent of the product line, as well as management commitment to support the product line approach. In terms of communication principles this helps to make people feel involved and convinced. In terms of the mind set it addresses the issue of fighting the *not-invented-here* syndrome and helps to improve the product line focus people have. It also raises the importance of the product line per se — as opposed to the individual product as the guiding principle for the developer.

In terms of communication patterns the above principles mostly focus on the aspect of involving practice which can be guided by the adequate structuring of the organization of work. But we identified additional aspects in the context of the communication patterns like the role of champions. Thus, we regard it as very important to:

- Get somebody in charge of the product line
  - this may be a managerial role *or*
  - a steering board with members from the different product units, which has at least management backing

Here, we actually know of cases where the mere assignment of a person to responsibility of the product line and that correspondingly the product line itself became an explicit entity lead to major changes in terms of the overall work and mind set.

- A clear product line vision (what can be future products) should be established, and management should be aware that it is measured based on its adherence to this vision and the corresponding messages it sends, both in wording and in actions.

This is actually of major importance, as from our experience, one of the major killers for institutionalizing a product line program in an organization is that the management sends, not by its words, but by its actions signals that lead the development to turn away from the product line and focus on individual products. Thus, this needs to be strongly avoided. Typical examples of such inappropriate management messages are: despite warnings from development personnel management is not willing to accept reasonable schedule slippage, but is rather willing to accept that development shortcuts are taken that lead to major compromises in the product line architecture. A similarly disastrous signal is to acquire projects at any cost, even if their required functionality does not fit the product line vision.

Finally, an important principle is that the people themselves should experience the benefits of product line development by experiencing that the overall work becomes more structured and disciplined and that their workload is reduced. Thus:

- It is important that saved effort through product line development is *not* completely compensated by management by the assignment of other tasks, but is rather used — at least in the medium time frame — to reduce the overall schedule pressure people experience.

In terms of organizational structures we were surprised to discover that the organizations do not necessarily adhere to the established principles of domain vs. application engineering units [4]. For example in the Owen case study [5] a domain engineering unit hardly exists. Architectural integrity is only assured by a steering board. On the other hand in the CelsiusTech case study [7] an application engineering unit hardly exists. Application engineering is only responsible for capturing the customer requirements. All development is performed in the domain engineering unit. Thus, product development can be performed successfully rather independently of the formal organization. However, as we discussed above in all the cases we regard as successful, the informal organization was structured according to the aforementioned principles.

While the success factors we presented here are derived in a more analytical manner, they are in good agreement with success factors derived from large-scale field studies [12].

5 Conclusions

In this paper, we discussed the aspect of people issues in transitioning to and continuously operating a successful product line project. In particular, we focused on the mind set of people and its relation to communication patterns and organization. While there is no strict one-to-one relationship, there is a dependency among these aspects. This dependency can — to some degree — be exploited in order to favor product line development. Even more important: a negative influence on one of them will usually influence the others. This can have a ripple-effect leading to strongly negative influences on an ongoing product line development.

We based our analysis of these issues on our experience in this area, but included existing case study reports. In some cases where product line development was unsuccessful, this had its roots not in technical problems, but rather in human issues, that were strongly related to the factors described here. In our discussion we derived some success factors we regard as key aspects in introducing product line development.

ACKNOWLEDGMENTS

We would like to thank all the people in different companies, I had the opportunity to work with over the course of the last seven years. Most notably of course, Martin Verlage from Market Maker Software AG, who also contributed valuable comments to this paper. I did also appreciate intensify exchange of ideas in the European ESAPS and Cafe projects. Special thanks go also to Paul Clements of the SEI for many interesting discussions and his thorough work on documenting case studies. Without this body of knowledge generalizable analysis like this one would be very difficult.

REFERENCES

1. D. Weiss and C. Lai. *Software Product-Line Engineering*. Addison–Wesley, 1999.
2. P. Clements and L. Northrop, *Software Product Lines: Practices and Patterns,* Addison Wesley Longman, 2001.
3. T. Wappler. *Remember the Basics*. In: Software Product Lines—Experience and Research Directions, Proceedings of the First Software Product Lines Conference (SPLC1), pp.73–84, Kluwer Academic Publishers, 2000.
4. J. Bosch. *Organizing for Software Product Lines*, In: Third International Workshop on Software Architectures for Product Families (IWSAPF-3), pp.126–143, 2000.
5. P. Toft, D. Coleman, and J. Ohta. *A Cooperative Model for Cross-Divisional Product Development for a Software Product Line*. In: Software Product Lines — Experience and Research Directions, Proceedings of the First Software Product Lines Conference (SPLC1), pp. 111–132, Kluwer Academic Publishers, 2000.
6. J. Dager. *Cummin's Experience in Developing a Software Product Line Architecture for Real-Time Embedded Diesel Engine Controls*. In: Software Product Lines — Experience and Research Directions, Proceedings of the First Software Product Lines Conference (SPLC1), pp. 23–46, Kluwer Academic Publishers, 2000.
7. *CelsiusTech — A Case Study in Product Line Development.* In: Software Architecture in Practice. L. Bass, P. Clements, and R. Kazman. Chapter 16 with L. Brownsword, Addison Wesley, 1998.
8. P. Knauber, D. Muthig, K. Schmid, and T. Widen. *Applying Product Line Concepts in Small- and Medium-Sized Companies.* IEEE Software, Vol. 17, No. 5, pp. 88–95, 2000.
9. P. America and J. van Wijgerden. *Requirements Modeling for Families of Complex Systems.* Third International Workshop on Software Architecture for Product Families (IWSAPF'3), March 15-17, 2000, Las Palmas de Gran Canaria, Spain. Springer LNCS 1951.
10. P. Clements, C. Gacek, P. Knauber, and K. Schmid. Successful software product line development in a small organization. In P. Clements and L. Northrop, editors, *Software Product Lines: Practices and Patterns,* chapter 11. Addison Wesley Longman, 2001.
11. K. Schmid, U. Becker-Kornstaedt, P. Knauber, and F. Bernauer. *Introducing a Software Modeling Concept in a Medium-Sized Company.* Proceedings of the 22th International Conference on Software Engineering (ICSE'00). pp. 558–567, 2000.
12. D. Rine and R. Sonnemann. *Investments in reusable software: A study of software reuse investment success factors*. Journal of Systems and Software. Vol. 41:1, pp. 17–32, 1998.
13. W. Lim. *Managing Software Reuse.* Addison-Wesley-Longman, 1998.
14. P. Knauber and G. Succi (Eds.), *Proceedings of the First Workshop on Software Product Lines: Economics, Architectures, and Implications,* IESE Technical Report 070.00/E, 2000.
15. P. Donohoe, Editor. *Software Product Lines: Experience and Research Directions* — Proceedings of the First Software Product Line Conference (SPLC1). Kluwer, 2000.
16. W. Mellis. *Process and Product Orientation in Software Development and their Effect on Software Quality Management.* M.J. Wieczorek and D.B. Meyerhoff (eds.): Software Quality — State of the Art in Management, Testing, and Tools. Springer, 2000.
17. M. Conway. *How Do Committees Invent*. Datamation, Vol. 14: 10 (October 1968): 28-31.
18. *The Product Line Practice Framework*, Version 4.1, http://www.sei.cmu.edu/plp/framework.html