

How Much Agility Do We Need? — A View From A Product Line Perspective

Klaus Schmid

Fraunhofer Institute for Experimental
Software Engineering (IESE), Sauerwiesen 6
D-67661 Kaiserslautern, Germany
Klaus.Schmid@iese.fraunhofer.de

1 INTRODUCTION

I do not consider myself an agile supporter; nor am I in favor of heavy-weight methods. The true question to me is not whether to be agile and which method is best, but rather which is the most appropriate approach in a given situation.

Four years ago, when we defined the PuLSE¹ approach to product line software engineering, we did not define it as a fixed method, but rather we intended it to be a very flexible and adaptable approach to product line engineering (cf. Figure 1, [3]). In particular, this method introduces the concept of a customization step; a step to define the actual method instance to be applied [4].

Product line engineering [1, 2] is an approach to systematically define a whole set of systems that are developed in a systematic manner while exploiting reuse opportunities on a large scale. How product line development is performed will differ from organization to organization. Actually, from an economic point of view, it is important to adapt the approach to the specific situation at hand [8]. However, it has been well demonstrated that this approach provides immense benefits (in particular in terms of cost) for software development.

The key idea of product line development is to build a set of assets that can be used as a generic platform for all systems an organization develops (cf. Figure 2). This development happens in the domain engineering phase of product line development. In particular in larger organizations this may even be a different suborganization. A key asset in this phase is the reference architecture. A common architecture that is to be shared by all products in the product line. Based on these common, product-line-specific components the final products are developed. This happens in the application engineering steps. In very large organizations, each of the various application engineering projects can be run by different organizations.

As a product line needs to deal with a number of products simultaneously, there is a strong need for systematicity in the overall process. Moreover, the need to continuously trade off between different products requires a strong discipline. For this (and some other) reasons, product line development is usually related with rather heavy-weight CMM-type development.

When we defined PuLSE, we defined it as strongly adaptable. Originally, we thought of it as being adaptable to a large number of organizational context factors like existing modeling methods, the distribution of the development team, the existing tool environment, the need to do safety-critical system, etc. In the mean-while we especially had the opportunity to work together with small and medium-sized companies and in particular small development groups in larger companies [7]. Especially in those cases we found a high degree of flexibility in the organizations, a virtue necessary to be preserved. Especially in those cases we found ourselves confronted with the question:

How much agility is compatible with product line development?

We will now discuss the interrelation of product line development and agile development values. As there is not a specific description of what agile development is, we will use the principles and values of the Manifesto for Agile Software Development as our basis [5, 6].

2 The Agile Principles From A Product Line Point of View

From the point of view of a product line guy, we can find good and bad — or perhaps better said: appropriate and inappropriate — in the agile concepts. The main part of the analysis is driven by the criterion that product line

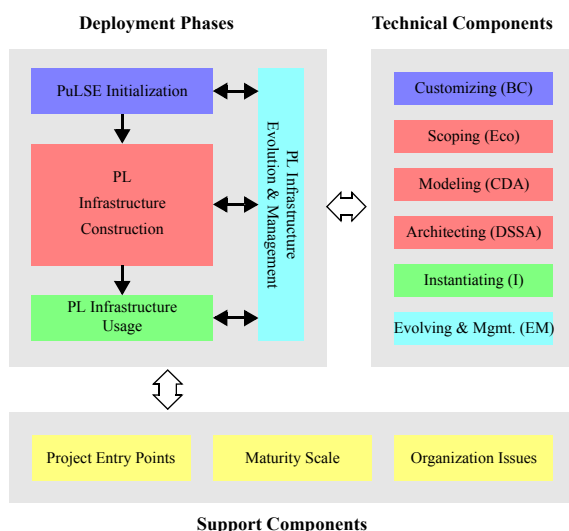


Figure 1: PuLSE Overview

¹ PuLSE stands for Product Line Software Engineering and is a registered trademark of the Fraunhofer IESE.

development is not focussed on a single customer, but rather on developing value through addressing a larger customer base, which, due to the incompatibility of some aspects can imply to leave some customer requirements unsatisfied.

2.1 Values of Agile Development

The values of agile development can be regarded as a cornerstone, a least common denominator of agile development:

- *Individuals and interactions over processes and tools*
The importance of individuals in setting up product lines is meanwhile more and more recognized; there is no well-running product line that has not some visionary person that made it start. However, there is a larger need for processes than with single-system development due to the fact that the complexity of software development is usually larger as more people are involved. Moreover, the separation of domain engineering and application engineering demands clear hand-over points.
- *Working software over comprehensive documentation*
Again we see the same picture; while the basic principle is compatible with product line development, it is shifted a bit towards the traditional style. Key here is the fact that even in cases where the individual product is short lived, the product line is *always* long-lived. It needs to be maintained over years and stand a considerable amount of people turn-over.
- *Customer collaboration over contract negotiation*
Here a serious shift happens in a product line situation, as the value of customer collaboration is definitely smaller. Moreover, a well-known problem in product line development is the “danger of the first customer”: focussing too strongly on the requirements of the first customer may endanger the product line vision right from the start.

While the customer is very important in a product line, focussing too strongly on his specific requirements may easily destroy the conceptual integrity of the product line and the resulting product line architecture will no longer scale to the needs of future products.

- *Responding to change over following a plan*

This is very similar to the previous one. Responding to change too strongly, especially if it is a single-customer need may easily dilute the concept of the product line. Moreover, changes done to the product line, which are triggered by one customer, may effect by virtue of the common assets other customers as well: an issue that needs to be addressed by adequate planning and analysis.

2.2 Additional Values of Product Line Development

Based on the above discussion we find that some aspects are different in a product line situation over the typical assumptions in agile projects.

- Usually requirements put forward by different customers are mutually exclusive. Thus they need to be thoroughly weighted in the product line development process:
Balancing the needs of the various customers is more important than focussing on a single products.
- Product line development is rather long-lived. This needs to be adequately addressed by the supporting processes and documentation.
Sustaining the product line is more important than getting the current product out of the door.
- Underlying to the product line there are conceptual principles that should be obeyed as otherwise the product line as a whole is at risk:

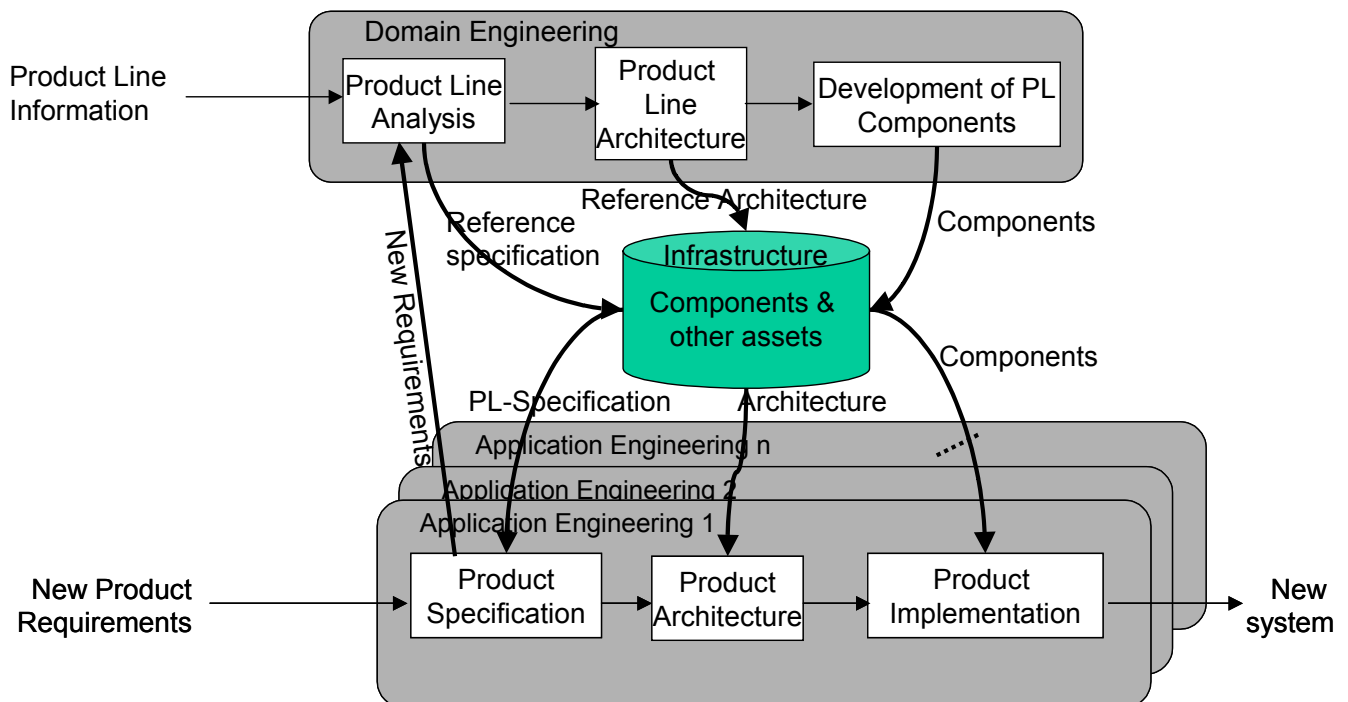


Figure 2: PL Engineering

Conceptual integrity of the product line is more important than technical beauty of a single product.

We can summarize these product line values by the following principle:

We value the *product line* higher than the individual *product*.

3 Conclusion

Product lines is somewhat different from single-system development, the usual working place of agile methodologies. This is reflected in somewhat different interpretations of the agile values. Moreover, the focus on product line development adds another value that needs to be addressed in decision making.

All this said, the core question remains: how much agility is needed in a product line situation. We answered this by focusing on values and trade-offs in the tradition of the agile manifesto. This leads us to look at the specific situation and perform a situation-specific trade-off.

This is very similar to the situation for which the PuLSE-BC approach was originally intended: for adapting a software development method. we see the possibility of using a similar approach to adapt the specific agile approach to the organization, perhaps introducing agile principles just to the right degree adapted to the situation.

REFERENCES

1. Paul Clements and Linda Northrop. *Software Product Lines*. Addison-Wesley. 2001.
2. Jan Bosch. *Design and Use of Software Architectures*. Addison-Wesley. 2000.
3. J. Bayer, O. Flege, P. Knauber, R. Laqua, D. Muthig, K. Schmid, T. Widen, J.-M. DeBaud. *PuLSE: A Methodology to Develop Software Product Lines*, Symposium on Software Reusability, 1999, ACM Press, pp. 122–131.
4. Klaus Schmid and Tanya Widen. *Customizing the PuLSE Product Line Approach to the Demands of an Organization*. Software Process Technology, 7th European Workshop, EWSPT'2000, LNCS 1780, Springer, 2000, pp. 221–238.
5. *Manifesto for Agile Software Development*. <http://agile-manifesto.org/>
6. Jonna Kaleremo and Jenni Rissanen. *Agile software development in theory and practice*. Master thesis, University of Jyväskylä.
7. 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, 2000, pp. 88–95.
8. Klaus Schmid and Martin Verlage, *The Economic Impact of Product Line Adoption and Evolution*, IEEE Software, Vol. 19, No. 6, 2002, pp. 50–57.