Preface

I. Software Product Line Engineering

Are you interested in producing software products or software-intensive systems at lower costs, in shorter time, and with higher quality? If so, you are holding the right book in your hands.

Higher quality, lower cost, and shorter development times

Software product line engineering has proven to be the methodology for developing a diversity of software products and software-intensive systems at lower costs, in shorter time, and with higher quality. Numerous reports document the significant achievements and experience gained by introducing software product lines in the software industry. Chapter 21 of this book summarises several cases.

Software product line vs. software product family

Concerning the terminology, there is an almost synonymous use of the terms "software product family" and "software product line". Whereas in Europe the term software product family is used more often, in North America the term software product line is used more frequently. This is, among other things, reflected in the names of the two former conference series (the software product line conference series, started in 2000 in the USA, and the product family engineering (PFE) workshop series, started in 1996 in Europe) which were merged in 2004 to form the leading software product line conference (SPLC) series.

In this book, we use the term *software product line*.

II. Readers of the Book

Intended readership

The book is for those people who are interested in the principles of software product line engineering. It elaborates on the foundations of software product line engineering and provides experience-based knowledge about the two key processes and the definition and management of variability.

We have written the book for practitioners, product line researchers, and students alike.

III. Book Overview

Framework for product line engineering

The book is organised according to our framework for software product line engineering, which has been developed based on our experience in product line engineering gained over the last eight years. The framework stresses the key differences of software product line engineering in comparison with single software-system development:

Two processes

a) *The need for two distinct development processes*: domain engineering and application engineering. The aim of the *domain engineering* process is to define and realise the commonality and the variability of the software product line. The aim of the *application engineering* process is to derive specific applications by exploiting the variability of the software product line.

Variability

b) *The need to explicitly define and manage variability*: During domain engineering, variability is introduced in all domain engineering artefacts (requirements, architecture, components, test cases, etc.). It is exploited during application engineering to derive applications tailored to the specific needs of different customers.

Among others, the book provides answers to the following questions:

- How can we save development costs and development time and at the same time increase the quality of software?
- How can we establish proactive reuse in software development?
- What is the variability of a software product line?
- What are the key activities and aims of the domain and application engineering processes?
- How can we document and manage the variability of a product line?
- How can we ensure consistency of the variability defined in different development artefacts like requirements, architecture, and test cases?
- How can we exploit variability during application engineering and thereby derive specific products from a common core?

Part structure

The book is divided into six parts:

Part I:	Introduction
Part II:	Product Line Variability
Part III:	Domain Engineering
Part IV:	Application Engineering
Part V:	Organisation Aspects
Part VI:	Experience and Future Research

Part I, Introduction, motivates the software product line engineering paradigm, introduces our software product line engineering framework, and provides an introduction into the example domain used throughout the book.

Introduction

Chapter 1 outlines the basic principles of product line engineering and its roots in traditional engineering.

Chapter 2 introduces our software product line engineering framework. It defines the key sub-processes of the domain engineering and application engineering process as well as the artefacts produced and used in these processes.

Chapter 3 provides a brief introduction to the smart homes domain from which examples are drawn throughout the book for explaining the introduced principles and concepts.

Part II, Product Line Variability, defines the principles of the variability of a software product line and introduces notations to document variability in all software development artefacts.

Variability

Chapter 4 defines the principles of variability of software product line engineering and introduces our orthogonal variability model, which we use throughout this book to document variability in the various software development artefacts clearly and unambiguously.

Chapter 5 defines how to document variability in requirements artefacts, namely textual requirements, features, scenarios, use cases, statecharts, and class diagrams.

Chapter 6 defines how to document variability in architectural artefacts, namely in the development view, the process view, and the code view of a software architecture.

Chapter 7 defines how to document the variability of component interfaces and the variability within the internal structure of components.

Chapter 8 defines how to document the variability in test artefacts such as test cases, test case scenarios, and test case scenario steps.

Part III, Domain Engineering, defines the key sub-processes of the domain engineering process. For each of the sub-processes we define the construction of the common (invariant) product line artefacts as well as the variability of the software product line.

Domain engineering

Chapter 9 introduces the principles of the product management sub-process within the domain engineering process. This sub-process mainly deals with topics related to economics and, in particular, to product portfolio management.

Chapter 10 defines the principles of the requirements engineering sub-process. It defines and illustrates the identification and documentation of common and variable features and requirements for the software product line.

Chapter 11 deals with the definition of a reference architecture for the software product line. It shows how product line commonality and variability are incorporated in the reference architecture.

Chapter 12 deals with the detailed design of reusable software components. It defines how the commonality and variability defined in the reference architecture is mapped onto components.

Chapter 13 discusses the influence of variability on the different test levels and presents and analyses test strategies with regard to their applicability in software product line engineering. The main focus is on establishing a systematic reuse of test artefacts in product line test activities.

Chapter 14 presents a technique for selecting commercial off-the-shelf (COTS) components, which takes into account the variability of the software product line. We consider components that provide a significant fraction of the overall functionality, the so-called high-level components.

Application engineering

Part IV, Application Engineering, defines the key sub-processes of the application engineering process. It shows how the orthogonal definition of variability established during domain engineering supports the exploitation and consistent binding of variability during application engineering – and thereby facilitates proactive reuse.

Chapter 15 defines the application requirements engineering sub-process. It tackles the problem of exploiting the common and variable artefacts of the software product line when defining an application. The chapter demonstrates how the orthogonal variability model supports the reuse of product line artefacts during application requirements engineering.

Chapter 16 deals with the application design sub-process which derives an application architecture from the reference architecture. By binding the variability according to the application requirements the required variants are selected and integrated into the application architecture. The sub-process also adapts the design according to application-specific requirements.

Chapter 17 deals with the realisation of a specific software product line application. Ideally, the realisation is achieved through a configuration of reusable domain components and application-specific ones by exploiting the commonality and variability of the components and their interfaces.

Chapter 18 deals with application testing. It shows how the variability – integrated into the domain test artefacts – supports the reuse of test case designs in application engineering. Consequently the effort of developing test cases for the different product line applications is significantly reduced.

Organisation aspects

Part V, Organisation Aspects, elaborates on two key aspects to be considered when introducing a software product line into an organisation: the organisation structure and the transition strategies.

Chapter 19 discusses the benefits and drawbacks of different organisation structures for software product line engineering.

Chapter 20 outlines transition strategies for moving from a single software production to a software product line. It discusses when to apply which strategy depending on the current situation within the organisation.

Experience and future research

Part VI, Experience and Future Research, reports on the experience with product lines and briefly describes several essential topics for future research.

Chapter 21 summarises experience reports about the application of the software product line engineering paradigm in several organisations. It also provides an annotated literature reference list as a guide for further reading.

Chapter 22 outlines key challenges for future research in the area of software product line engineering.

End of the book

In addition, we provide at the end of the book:

- Information about the *authors*
- The *literature references* used throughout the book
- A *glossary* for software product line engineering
- The *index*

IV. Share Your Experience!

Feedback

We are interested in your feedback. If you have any suggestions for improvements, or if you have detected an error or an important issue that the book does not cover, please do not hesitate to contact us at:

SPLE-Book@software-productline.com

or visit the book web page:

www.software-productline.com/SPLE-Book

V. Acknowledgements

Eureka/ITEA, public authorities

We would like to thank Eureka/ITEA, BMBF (Germany), NOVEM/ SENTER (Netherlands), and all the other national public authorities for funding the projects ESAPS (1999–2001), CAFÉ (2001–2003), and FAMILIES (2003–2005). Most of the results presented in this book have been researched and validated in these projects.

Layout and consistency

Our special thanks go to Ernst Sikora who helped us significantly in improving the consistency and the layout of the book, and to Silja Recknagel for her support in polishing the English.

Publisher

We thank Ralf Gerstner, Ronan Nugent, and Ulrike Stricker from Springer-Verlag, Heidelberg, for their support in getting this book to market!

Collaborators and colleagues

Last but not least, our thanks go to our collaborators within industry and universities, at both the national and the international level. Without the insight gained in numerous discussions, this book would never have been possible.

Klaus Pohl — University of Duisburg-Essen, Germany
Günter Böckle — Siemens Corporate Technology, Germany
Frank van der Linden — Philips Medical Systems, The Netherlands

May, 2005