Omega 41 (2013) 536–552

Contents lists available at SciVerse ScienceDirect

Omega

journal homepage: www.elsevier.com/locate/omega

Integrated production strategy and reuse scenario: A CoFAQ model and case study of mail server system development

Jiafu Tang [a,*], Zhiqiao Wu [a,b], C.K. Kwong [c], Xinggang Luo [a]

[a] Department of Systems Engineering, State Key Lab of Synthetic Automation of Process Industry, Northeastern University, Shenyang 110004, P.R. China
[b] College of Business Administration, Shenyang University, Shenyang 110004, P.R. China
[c] Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon Hong Kong, P.R. China

ARTICLE INFO

Article history:
Received 7 February 2011
Accepted 17 July 2012
Processed by B. Lev
Available online 2 August 2012

Keywords:
Software reuse
Integrated decision
Optimization
Heuristic

ABSTRACT

One of the core problems in software product family (SPF) is the coordination of product building and core asset development, specifically the integration of production strategy decision and core asset scenario selection. In the current paper, a model of Cost Optimization under Functional And Quality (CoFAQ) goal satisfaction constraints is developed. It provides a systematic mechanism for management to analyze all possible products and evaluate various reuse alternatives at the organizational level. The CoFAQ model facilitates decision-makers to optimize the SPF development process by determining which products are involved in the SPF (i.e. production strategy) and which reuse scenario for each module should be selected to implement the SPF toward minimum total developing cost under the constraints of satisfying functional and quality goals. A two-phase algorithm with heuristic (TPA) is developed to solve the model efficiently. Based on the TPA, the CoFAQ is reduced to a weighted set-covering problem for production strategy decision and a knapsack problem for the reuse scenario selection. An application of the model in mail server domain development is presented to illustrate how it has been used in practice.

1. Introduction

Similar to software product line engineering, *software product family* (SPF) development provides various kinds of software products to meet the requirements of several market regions simultaneously. *Products* belonging to an SPF are assembled from the organization's *core assets*, with each core asset having an attached process that describes how the asset is used [7,1]. SPF development is composed of a series of management activities that describe how the attached processes work together to yield products, and it captures how the organization builds any product in SPF. It is more complex than a typical project with single-product development. As shown in Fig. 1, the processes are composed of two important phases, namely, the production strategy development phase and the core asset development phase, conducted through *product building* and *core asset development,* respectively. In the first phase, product building contributes to an SPF from the perspective of domain market and recognizes common and unique requirements for each market region. Afterwards, production strategy is developed to identify variants (i.e., various products) that will give a high return of investment by saving costs in development, satisfying special market regions, and gaining a larger market share. In the second phase, the task of core asset development is to decide which core assets are developed and how to implement these core assets under different reuse scenarios based on the production strategy. Thus, core assets are assessed to determine the appropriateness of their inclusion in the SPF. Moreover, reuse scenarios of each core asset are identified to describe how the core asset is implemented in the SPF.

As implied in Fig. 1, the decisions in the two phases are correlated, and thus should be made in an integrated manner. Traditionally, the management activities in the two phases are conducted separately, which results in a local optimal solution from the system perspective. Each phase has different targets in its own scope rather than at the organizational level. Thus, learning how to coordinate effectively the two phases in SPF development becomes an important issue. The current paper aims to develop an optimization model to coordinate the production strategy and reuse scenario. A state-of-the-art solution is discussed first.

In relation to product building, a domain analysis is conducted to determine the kind of products to be built based on product-line architecture for a specific developer. The domain analysis guides a developer in selecting variants to be accommodated in an application. A feature-oriented analysis may, of course, be an

* Corresponding author. Tel.: +86 24 83673810.
E-mail addresses: tangjiafu@ise.neu.edu.cn, jftang@mail.neu.edu.cn (J. Tang).

0305-0483/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.omega.2012.07.003

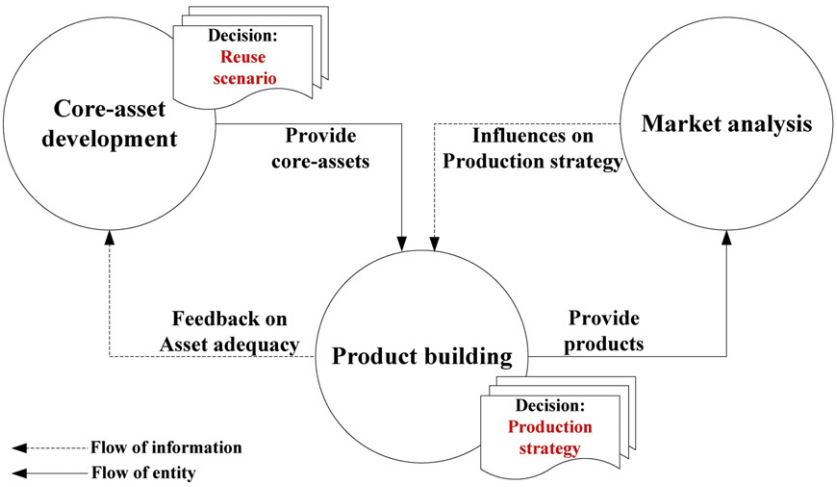

Fig. 1. Roles of core asset development and product building.

acceptable method for modeling common and variant requirements for product family [23]. As an output of feature-oriented analysis, a feature model is a very concise taxonomic form, the features of which are modeled as symbols [11]. However, the feature models cannot determine non-functional requirements explicitly, e.g., quality. Many researchers and practitioners have proposed goal-oriented models to identify variability at the early requirements of SPF [22,21]. Goal-oriented requirements engineering [19] is widely used for explaining and evaluating non-functional and quality requirements with a high degree of accuracy [2,4,20]. The overall satisfaction of non-functional and quality requirements depends on the performance of software architecture and components, indicating that total quality control can take place in the core asset development process.

Core asset development serves as a guide to product building in relation to how the assets should be used by attaching a process to each core asset. Sufficient information about each core asset is provided to facilitate the product builder's understanding about the assets and to make informed choices based on the total quality requirements of an SPF. However, such process may result in selecting several candidates or a particular asset. It may also result in finding that some desired functionalities cannot be found in existing core assets and that the functionalities should be implemented in a new component or asset. As indicated by Tomer et al. [26], systematic reuse of a core asset is not merely a technical issue. On the contrary, it is widely accepted as an organizational challenge. The activities in core asset development should be systematically measured or estimated, and the alternative assets should be evaluated and compared to support the entire reuse process effectively. Many reuse strategy selection models [24,25] have been introduced to increase systematically the efficiency of core asset reuse. Recently, the authors [27] have developed a systematic framework of core asset development process to support cost analyzing, comparing and choosing particular reuse scenarios such as build or buy, and white-Box reuse versus black-Box reuse. These works enable core-asset development to analyze and evaluate alternative reuse scenarios in the requirement phase in a precise manner.

In fact, SPF development may have significant consequences on the total costs of product builders and core asset developers. Therefore, the tools that support decisions to coordinate the efforts of product building and core asset development can be very helpful to the tasks of SPF practitioners. Specifically, as the main expenditures in SPF development, production strategy determination and core asset reuse scenario selection should not be considered separately, but rather jointly, to achieve cost savings.

Software reuse-based development essentially entails a type of multi-stage optimization problem. A significant breakthrough in this field is the first emergence of optimization techniques [5,16], rapidly drawing the attention of many scholars [9]. Jung [16] initially proposed a 0–1 knapsack model to select a set of software requirements to yield a maximum value while minimizing costs. Considering reliability growth as a function of cost, Berman and Cutler [5] introduced an optimization model to maximize the reliability of an assembly of assignments under a budget constraint. Thereafter, many scholars have considered it a cost-minimization problem within a reliability constraint. Taking delivery time and product reliability into account, Cortellessa et al. [10] developed a model that views it as a cost-minimization problem under quality constraints by building costs and quality attributes on a common set of decision variables in a special framework. These solutions are integrated for cost management and failure rate control in software development. Cortellessa et al. [8] further induced a goal-oriented requirements engineering and made optimal tools sufficient to describe precisely the satisfaction of system requirements. Goal-oriented requirements engineering is adopted in the present study to ensure the satisfaction of quality requirements under constraint. These approaches indicate that a pre-selection of components and products at the requirements phase, which is associated with the combination of functional and quality requirements, can sensibly improve the efficiency of the entire development process [14] and reduce development costs [8]. However, none of the previous works provides an optimization method to support full SPF development life cycle, i.e., simultaneous optimization of products and reusable component selection. Automatically accomplishing it in an organization remains a difficult task. There is a complex and nonlinear relationship between the two factors. The models or algorithms for product family reuse optimization considering the two aspects of cost simultaneously have received little attention. The current paper attempts to address this unsolved problem, which broadly covers the work range of product building and core asset development using the optimization technique.

In the current paper, by introducing 0–1 variables to represent optional products and alternative reuse scenarios, and by combining integer variables to represent the number of reuses with the core-asset, the integrated problem addressed here attempts not only to select the most cost-effective products but also to show how to implement each core asset for the selected products. A model of Cost Optimization under Functional and Quality