Business Family Engineering: Does it make sense?

Ildefonso Montero, Joaquín Peña, Antonio Ruiz-Cortés

Departamento de Lenguajes y Sistemas Informáticos

Av. Reina Mercedes s/n, 41012 Seville (Spain)

University of Seville

{monteroperez, joaquinp, aruiz}@us.es

Abstract

Nowadays most companies in whichever field have a software system that helps managing all the aspects of the company, from the strategic management to daily activities. Companies are in continuous evolution to adapt to market changes, and consequently, the Information Technology (IT) infrastructure that supports it must also evolve. Thus, software companies are currently supporting this evolution with *ad hoc* techniques.

We think that, as it is being done for traditional software systems (non-oriented to business process) in the software product line (SPL) field, institutionalized techniques for performing a systematic reuse of business processes across different businesses can be introduced.

In this paper, we explore the feasibility of adapting SPL techniques, oriented to reuse software, to Business-Driven Development (BDD), oriented to reuse processes, across different businesses; we call this approach Business Family Engineering (BFE). As a result of our study, we show some of the problems we have identified and some of the key aspects needed to enable this new field.

1. Introduction

Nowadays most companies in whichever field have a software system that helps managing all the aspects of the company, from the strategic management to daily activities. Companies are in continuous evolution to adapt to market changes, and consequently, the Information Technology (IT) infrastructure that supports it must also evolve. Thus, software companies are currently supporting this evolution with *ad hoc* techniques.

Research fields such as autonomic computing (by means of self-* properties) or policy-based management, try to provide solutions for the evolution. Business-Driven Development (BDD) is another research field, which is the focus of this paper, that tries to solve this problem designing software systems starting from the business processes of the companies.

Business processes are designed to be executed over a process engine. Of course, current process engineers redesign the processes every time that is needed using ad hoc techniques to maximize the level or reuse from one version to another. In addition, when dealing with several businesses in a certain domain, many common features can be found, and reuse across businesses is also exploited.

There exist a field called software product lines (SPL) that systematizes the reuse across the set of similar products that a software company produces. We think that such systematization can be also applied in BPE improving the results achieved by current *ad hoc* techniques.

Clemens *et al.* defines in [3] *Software Product Line* (SPL) as follows: *a set of software-intensive systems, sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way*. The main goal of SPL approach is to obtain a reduction of the overall development cost and time for the products derived from the product line based on reuse. Basically, in SPL we obtain a set of software systems, called products. Each product contains common functionalities,

called features, and a set of specific features that differentiates one product from another.

The idea of applying SPL to BDD, has been explored by Schnieders *et al.* who define in [8] *Process Family Engineering* (PFE) as *a modern software development approach, which allows for the rapid and cost-effective development and deployments of customer tailored business process oriented systems*. Basically, PFE follows the SPL philosophy for managing the evolution of the business process of a unique business (manage only one software system). That is to say, each product in PFE represents an evolution of the process (at runtime).

Thus, PFE may be the solution to manage the evolution of the business process of a company, but to the best of our knowledge, there not exists an approach to build a product line of BDD systems.

In this paper, we expose the main concepts of the approach needed to build a product line of businesses, that we call *Business Family Engineering* (BFE). In addition, from our analysis, we conclude that PFE is useful for managing single businesses, but it is not feasible for a set of businesses (BFE). We expose these limitations concluding that this approach can be used to manage the evolution of each business in a BFE.

This paper is structured as follows: Section 2 presents the background needed about SPL and PFE proposals; Section 3 presents the main differences between SPL and PFE; Section 4 presents the main features of BFE; Section 5 presents a discussion about BFE as a realistic solution in the scope of SPL for PFE systems; and finally, in the last section, we draw the main conclusions and the future research lines needed to enable a business process family infrastructure.

2. Preliminaries

2.1. Software Product Lines

Pohl *et al* define in [5, 6] that SPL development aims at and achieves pro-active, constructive reuse, based on the idea to develop software products which share a significant amount of features based on a common platform. The SPL approach is devoted to overcome complexity providing all the techniques needed for enabling the mass production of software in a certain application domain. The variability concept appears in SPL to represent the differences and commonalities inside an application domain. Variability is one of the critical aspects of SPL and it must be managed at all the stages of SPL development.

The software process of SPL is divided into two main stages: *domain engineering*, which is in charge of providing the reusable core assets that are exploited during the derivation of products, done at a second stage named *application engineering* [6].

One of the most accepted techniques to represent the set of products of a SPL are *feature models* [4]. The main goal of feature modelling is to identify commonalities and differences among all products of a SPL. A feature model is a compact representation of all potential products of an SPL showing a set of features in an hierarchical structure where it is shown which features are mandatory, optional or alternative. In Figure 1 is shown an example of an E-shop feature model, where there are three mandatory features: *Products*, *Shopping Cart* and *Checkout*. It represents that all the products of the SPL represented by this feature model must have the features catalogued as mandatory.

2.2. Process Family Engineering

Process Family Engineering (PFE) is an approach given by PESOA research group from Hasso Plattner Institute for IT Systems Engineering in [1]. In the same way that SPL approach provides all the techniques needed for enabling the mass production of software in a certain application domain, PFE approach provides all the techniques needed for enabling the mass production of processes in a certain business. Each product represents a set of processes enabled at a certain moment of the execution. In PFE we obtain only one software system, where the features are processes, and where this system envolves at runtime. Every evolution of the process represents a product that contains a subset of all features. However, the systems itself contains all the features of the family.

The main tool for representing the set of processes contained into a business are feature models, and the tool for representing an specific

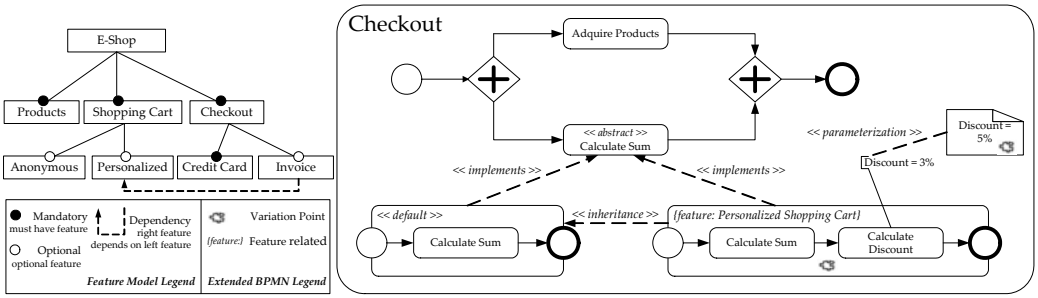

Figure 1: Example of feature model and extended BPMN by PFE approach

process is *Business Process Model Notation* (BPMN). It is defined by OMG in [2] as a flow chart based notation for defining business processes. BPMN provides (i) a graphical notation based on *Business Process Diagram* (BPD), which is a diagram used to design and manage business processes; and (ii) a formal mapping to an execution language: *Business Process Execution Language* (BPEL). PESOA introduces an extension of BPMN to represent variability in a process [7].

Figure 1 shows an example of a feature model of an E-shop business and an extended BPMN to represent a checkout process. Feature model represents all the processes contained into the E-shop business, if a process is denoted as *mandatory* it must be present in all the possible configurations of the business, for example: *Checkout*. Each process is represented using BPMN with the extensions proposed by PESOA. As shown in Figure 1 variation point extension is represented as a puzzle-piece graph notation and for feature and processes relationship we see that *Calculate Sum* can be implemented as a sequence of *Calculate Sum* and *Calculate Discount* subprocesses that is applied when the feature *Personalized Shopping Cart* is selected.

3. Main differences between SPL and PFE

In SPL a product is composed of a set of common features and a set of variable features. Common features appears in all products and variable features appears under demand of products's consumers. Observing a certain product of a SPL, although it is described as a set of fixed features, some features can be in use in a certain moment and some not. Thus, in SPL the evolution of the system at runtime is not taken into account in the feature model. In PFE each feature is a process and all of them appear into the product, but at runtime there exists a set of products based on selection of features/processes.

As can be observed in Figure 2, where we depict how SPL and PFE products are generated, SPL products are implemented by software artifacts that for each of them there exists a feature selection phase that generates the final products (a set of core and variable features). PFE products are implemented by processes that for each of them there exists an evolution in execution time incrementing or decrementing the variable set of features. Each product is a software system based on processes.

4. Business Family Engineering

In this section, we define the main aspects of BFE.

4.1. BFE Definition

Business Family Engineering (BFE) can be defined as: *a set of software systems driven by business processes (hereafter business) where each product of the family has a set of common processes and a set of variable processes*. The formal definition of BFE can be represented as follows: Let *BF* be a Business Family that is a set of $n > 0$ businesses

$$BF = \{B_1, B_2, \ldots, B_n\}$$

where each *B* represents a business. Each business *B* is a set of processes (denoted with *P*). Thus, each B_i in *BF* can be defined as follows:

$$Bi = \{P_1, P_2, \ldots, P_k\};\ k > 0;\ 1 \leq i \leq n$$

Given this it holds that there exists a set of common processes between whichever set of businesses. Let B_i and B_j be two businesses contained in *BF* where $1 \leq i, j \leq n$:

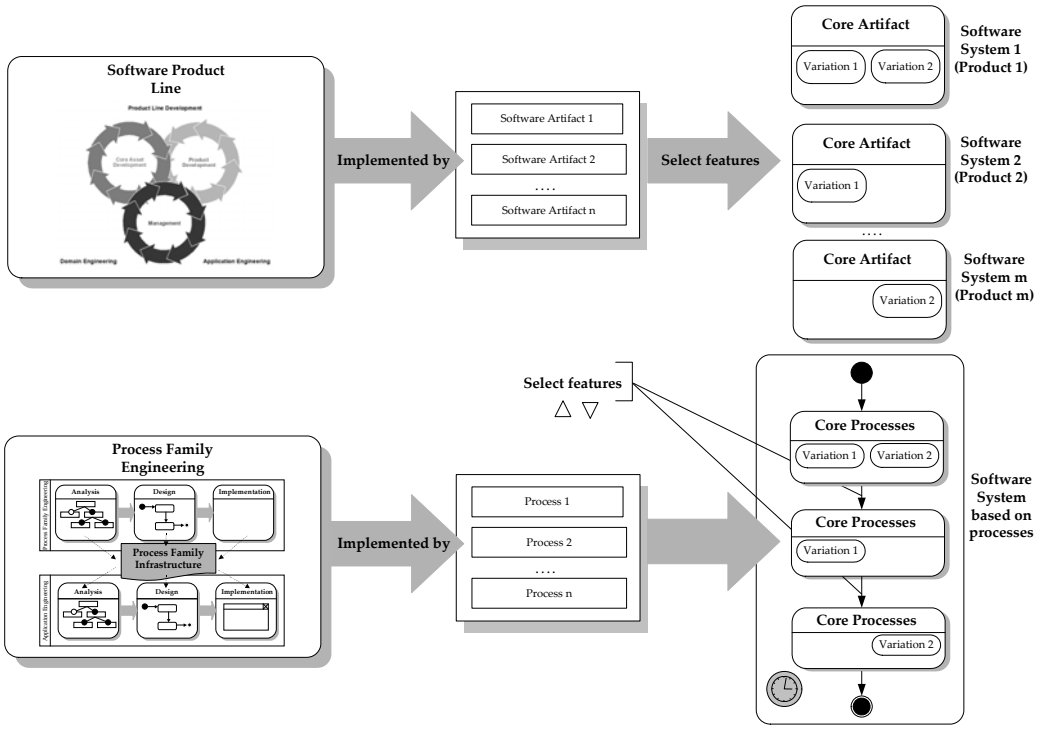

Figure 2: SPL and PFE approaches

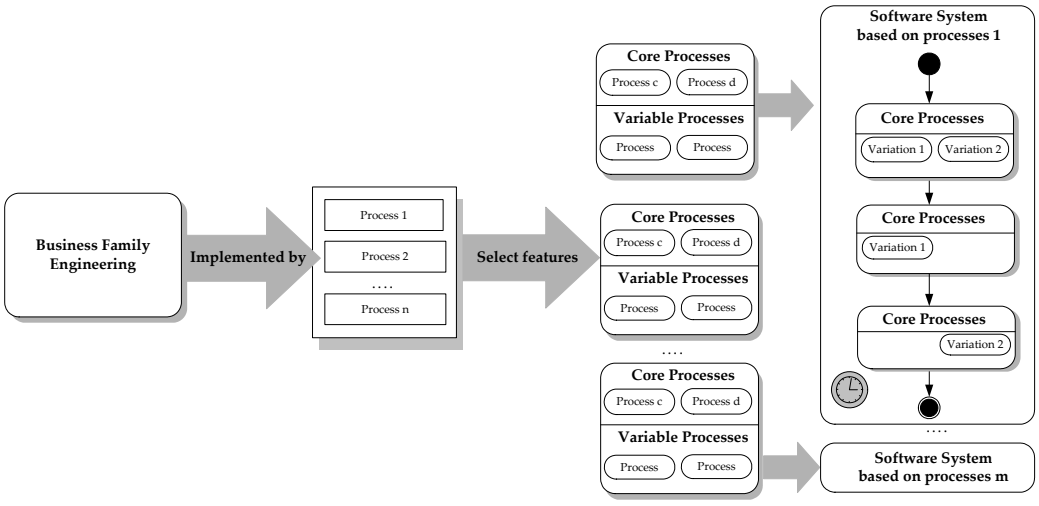

Figure 3: BFE approach

$$B_i \cap B_j \neq \emptyset$$

Thus, we can say that a business family can be also defined as a set of core and variable processes/features. Let *CF* be the set of common processes or features and let *VF* be the set of variable features, *BF* can be defined as a tuple *(CF, VF)* as follows:

$$BF = (CF,\ V\ F)$$

In that way, a business B_i is defined formally as a tuple containing all the *CF* and a subset of *VF* denoted as *SVF:*

$$B_i = (CF,\ SVF \in VF\ )$$

4.2. Integration of BFE with PFE

PFE provides techniques to manage the evolution of the business process of a company based on SPL ideas and BFE provides a SPL of BDD systems. In this section, we present our first steps towards the integration of both approaches.

Figure 3 depicts the integration between BFE on PFE. As shown, each business contains a set of core processes, *CF*, and a set of variable processes, *VF*. However, in PFE the processes/features appear and disappear at runtime. As shown before, each configuration of the set of processes enabled at a certain moment represents a product. Thus, we can say that the *CF* of a *BF* are always enabled at runtime, but the set of processes in *VF* is not fixed at runtime.

Thus, as *PFE* defines, we can set up a product line that takes into account this runtime variability. For formalizing these concepts we should redefine each business *B* of a *BF*.

$$B = (CF,\ SVF \in\ VF,\ F\Delta\ :$$

$$:\ t,\ \{Feature \times \ldots \times Feature\} \mid\rightarrow$$

$$\mid\rightarrow \{Feature \times \ldots \times Feature\})$$

where F_Δ is a function that given an instant *t* transform the set of SVF_t into the new set of variable features of the following time instant *t+1*, that is to say SVF_{t+1}, formally:

$$F_\Delta\ (t,\ SVF_t) = SVF_{t+1} \in\ VF$$
$$\bullet\ SVF_t \neq SVF_{t+1}$$

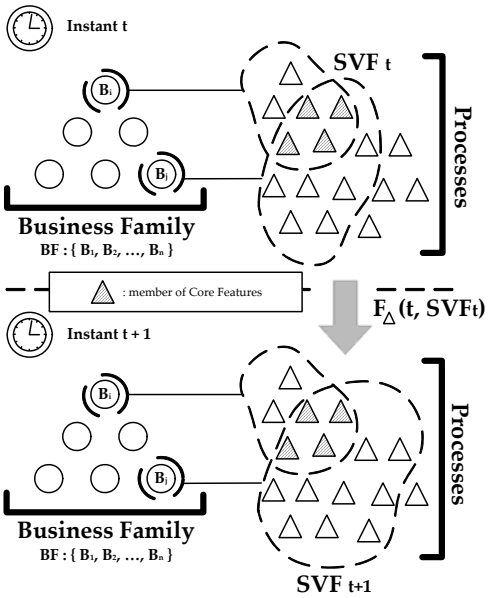

Figure 4: Evolution of a business into BFE

Figure 4 sketches a graphical representation of *F*Δ, where it is represented the transformation of SVF_t into SVF_{t+1}. In an instant *t* there exists an specific set of SVF_t for business B_j that evolves in instant *t+1* to another different set SVF_{t+1}. The evolution is defined by the *F*Δ in *t*.

5. Discussion

In this section, we conduct an analysis about the main problems identified in BFE approach. The first problem is about combinatorial explosion. The main reason about this problem is that BFE consists on building a product line of PFE product lines. Thus, there exists a SPL that grows very rapidly.

The second problem is about using feature models to represent process changes on runtime. Feature model are designed to represent design time variability, thus they are not adequate for runtime variability.

In Figure 5 we present a case study about a *Restaurant Chain*, that uses feature model to

represent different products. Pay attention on *Restaurant PNIS07* feature model. There exists a process called *Serve* that depending on the moment can be *Serve Fast* or *Serve Normal*, but feature model is not expressive enough for representing dynamic evolutions on runtime since it does not support runtime variability.

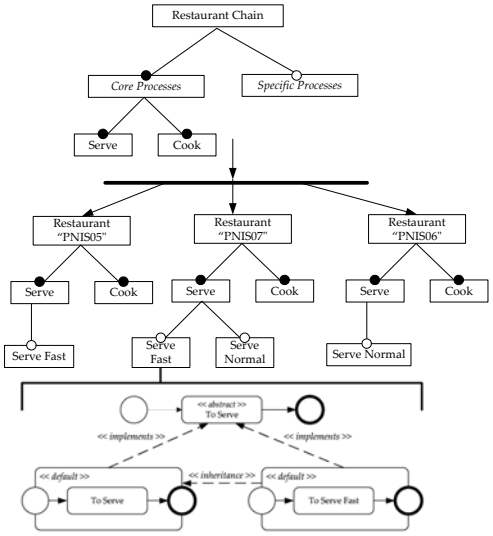

Figure 5: Case Study: Restaurant chain

Possible solutions to the identified problems are:

- For combinatorial explosion: using one feature model to BFE and another feature model for each PFE, obtaining an hypercube structure, that represents all the possible variations of products.
- For documenting dynamic evolutions: using one feature model to BFE and another feature model for all the possible products introducing an extension of feature model ables to represent runtime variability and that aglutinates the variability of all PFE products. Thus we may solve both identified problems.

6. Conclusions

The main conclusion of this paper is that BFE is feasible. Its main benefit is that software companies that provide BDD solutions, can reuse process building a product line where a set of common processes is extended with the processes needed for each customer in a systematic way, thus reducing costs (in time and money) and improving the quality of their products, since they are tested for several clients.

Another important conclusion is that PFE cannot be used directly for BFE. PFE provides techniques to manage the evolution of the business process of a company based on SPL ideas, however all the variable features of the process are added to the final software system, enabling or disabling them at runtime. While BFE needs techniques that allow adding only those features that the customer requires. In addition, techniques used in PFE presents drawbacks. Mainly, feature models are used to represent runtime variability, while these models are devoted to static variability.

As PFE is quite valuable for runtime variability, we conclude that BFE must be integrated with PFE, but a number of problems arise. Mainly, as a result of having a product line (BFE) of product lines (PFE) it occurs an state explosion that hinders the feasibility of this approach.

Acknowledgements

This work has been partially supported by the European Commission (FEDER) and Spanish Government under CICYT project Web-Factories (TIN2006-00472).

References

[1] J. Bayer, W. Buhl, C. Giese, T. Lehner, A. Ocampo, F. Puhlmann, E. Richter, A. Schnieders, J. Weiland, and M. Weske.Process family engineering. modeling variant rich processes. Technical report.

[2] BPMI. Business process modeling notation (BPMN) version 1.0 - may 3, 2004. OMG.

[3] P. Clements, L. Northrop, and L. M. Northrop. Software Product Lines: Practices and

Patterns .Addison-Wesley Professional, August 2001.

[4] K. Czarnecki and M. Antkiewicz. Mapping Features to Models: A Template Approach based on Superimposed Variants. 2005.

[5] G. Halmans and K. Pohl. Communicating the variability of a software-product family to customers. Inform., Forsch. Entwickl., 18(3-4):113–131, 2004.

[6] K. Pohl, G. Böckle, and F. van der Linden. Software Product Line Engineering: Foundations, Principles and Techniques. Springer, September 2005.

[7] F. Puhlmann, A. Schnieders, J. Weiland, and M. Weske. Variability mechanisms for process Models. Technical report.