Business Case for a Product Line of Legacy Application Data-Middleware

Marcelo López H.
mlopez@dcc.uchile.cl

M.Cecilia Bastarrica
cecilia@dcc.uchile.cl

DCC, Universidad de Chile
Blanco Encalada 2120
Santiago, Chile

1. Abstract

Legacy applications represent software solutions for many organizations and businesses. These applications have been implemented using different IT platforms and few of these systems have been standardized or migrated to newer versions. Thus, there are a lot of heterogeneous applications running in different platforms, even within one organization. The need for interchanging strategic information between organizations or legacy applications is more common than a few years ago. Data interchange among these heterogeneous legacy systems is usually a major project. The solution is not unique and there might even be many solutions for every pair of legacy applications. We call data-middleware a product for interchanging information between two legacy applications. In order to develop a reusable and non-legacy implementation dependent solution, this product could be developed using the software product line paradigm. The development of this product as part of a software product line includes new practice areas that must be defined in a Business Case. A Business Case (BC) is a tool for making a business decision, because it predicts the organizational consequences of this decision. We describe a BC with three core practice areas: how the organization should be structured, the base architecture as the main initial asset, and how the data-middleware product line organization is launched and institutionalized.

Keywords: Software product lines, business case, data middleware.

2. Introduction

Most of the organizations have spent a lot of resources to build and keep an information technology platform as well as systems to support their businesses, in order to incorporate an advantage with respect to competitors.

During the last years many legacy applications have been developed using different operating systems, programming languages, paradigms, software engineers with different backgrounds, not to mention hardware and the new concepts involved in the Internet-based application development. In this scenario, there is a growing need for interchanging information among these heterogonous applications regardless their implementation and localization.

We call data-middleware a product that provides a common set of standards and technology to support interfaces and data interchanging between applications. Data-middleware should consider key functions and support procedures to be an adequate tool for managing and controlling data interchange between a pair of legacy applications.

The design, development and implementation of a data-middleware is not an easy task. Provided that, there are many opportunities for applying a data-middleware product, it is convenient to design a reusable solution. This article shows why the development of a data-middleware based on the software product line (SPL) paradigm should be considered a viable alternative. Following SPL implementation guidelines [6], we present a Business Case to show why a data-middleware product should be developed using a SPL framework.

A Business Case (BC) is a tool for making a business decision, because it predicts the organizational consequences of this decision [7]. Following this BC definition, we propose three major aspects to be considered: how the organization should be formed, the base architecture as the most important initial asset, and how the product line organization is launched and institutionalized.

The framework for a software product line, definitions and guidelines are presented in [6]. In [7] we also find a tool to decide whether it is convenient to use the software product line approach; this tool is called a Business Case. We have experience designing and implementing ad hoc solution for data middleware, and we realized there is a big opportunity for large scale reuse in this area. We reported part of our experience in [3].

2.1. Paper Overview

In Section 3 we describe generalities of the software product line approach. Section 4 presents the Business Case, our main contribution. We here include the organizational structure, the base architecture and the launching and institutionalizing of the data middleware product line. Finally, in Section 5 we describe part of our ongoing work and some of our conclusions.

3. Software Product Line Overview

A software product line (SPL) is 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 [7].

The principal SPL goal is obtaining substantial production economies when applications in a product line are developed from a common set of assets, in contrast to being developed separately, from scratch, or in an arbitrary and ad-hoc fashion.

SPL is a new paradigm for developing software products [6]. In essence, SPL involves *core assets development* carried out by the *Domain Engineering* and *product development* using the core assets implemented by the *Product Engineering*, both under the control of an organizational *Management* [7].

As we mentioned before, SPL is a new idea and new organizational management practices are required for the implementation of the entire product line effort. A Business Case (BC) is a tool that involves the definition of specific new practice areas that require preparation, planning and then execution and implementation. BC helps to decide to make a business decision for pursuing this opportunity or approach.

4. Global Business Case

4.1. Objective

The goal of our business case is to decide whether to develop a tool – called data-middleware - that supports the interchange of data between two legacy applications, regardless of their implementation using the software product line approach. Legacy implementations include different operating systems, software and hardware platforms, data structures and paradigms. The product line involves the generalization of key data-middleware functions and procedures in order to be reused in future products.

4.2. General Organization

Product line systems are developed and managed according to a life cycle that differs from the one used in an *ad hoc* development [7]. SPL organization must take the following responsibilities:

- Ensure that new products reuse the core asset base according to the production plan.
- Interact with the domain engineering to develop and evolve new capabilities of the core asset.
- Negotiate requirements of customers to discuss if new products fit within the scope of the SPL.

Bosch proposes five different models for a software product line organization [4]. The organization for producing the SPL of data middleware should be structured based on one of these five models. We present a brief definition of each model.

- Development department. In this model all software tasks are concentrated in a single unit. This model appears in small organizations (no more than 30 people) and those that provide consulting services. The principal objective of this unit is providing support to the rest of the organization. The key concept in this organization is that each member of the development team takes part in the domain engineering tasks as well as in product engineering duties.
- Business unit. In this model there are as many organizational units as assets (the scope definition, the base architecture, core specific components, etc). The essential idea is to develop a shared core asset community. It is estimated that this kind of model could apply to organizations with 30 to 100 employees. There is an obvious risk in this model that is that a business unit will focus on its own product(s) or assets first and the product, as an integrated application and the software product line as a whole, will only take a second place.
- Domain engineering unit. In this model, a special unit called domain engineering has the responsibility for developing and maintaining the core asset base. The product engineering units build the products using those assets. There may be as many product engineering units as different products in the SPL. Bosch indicates that this model could be applied when the organization exceeds 100 employees.

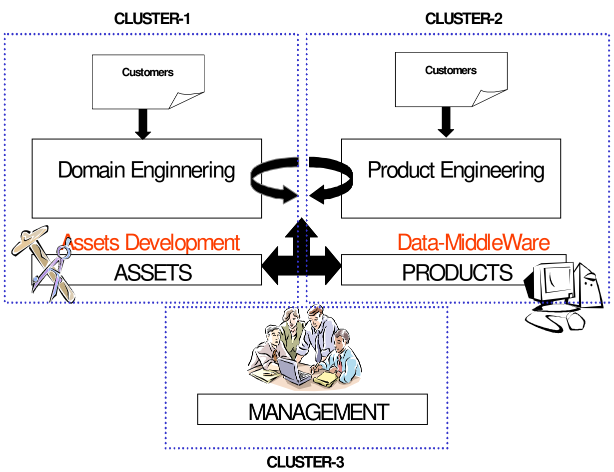

Figure 1 Legacy Application Data Middleware Global Organization

- Hierarchical domain engineering units. This model is applied to those organizations with a very complex or large structure. That is, the product line organization consists of a series of subgroups that have more in common with each other than other members of the product line unit. Each subgroup is in charge of developing specialized products of the line. Specialization is based on some subject, architecture or product line components.

- Organizational structure for a reuse business. Jacobson et all [9], called this kind of model a set of competence units, which contains workers with similar competencies and entity object types that these workers are responsible for. Jacobson defined the following SPL object types:
    - Requirements capture unit
    - Design unit
    - Testing unit
    - Component engineering unit
    - Architecture unit
    - Component support unit

Organization for data-middleware development using SPL paradigm could be structured following the domain engineering unit model proposed by Bosh.

Figure 1 proposes the general organization for developing data-middleware products, following SPL development [6][7]. This organization should consider at least one group of engineers for the domain area and one group for the product development. Management group also should be present in every SPL organization. This organization is the target for a well-established SPL company, but it is not necessarily there for the first product development.

There are two possible scenarios for developing the first product of the SPL: the first one where there is no prior product and no assets, so everything is new, and another scenario where there are some assets that are mined [10] from prior platform dependent applications. In both cases, the Domain Engineering group has almost all the work: either developing the assets or mining them.

There are two kinds of customers who will interact with the development organization, one that behaves as the initial customer or the customer of the first product of the line, and the other is a customer of a product of the well established SPL. These two groups have a clearly different relationship with the SPL organization. The first customer group participates in the construction of the first product: they provide definitions and requirements and the domain engineering collects, generalize and implement them. The second group of customers needs a product, but they can negotiate some variations according to the available assets; this

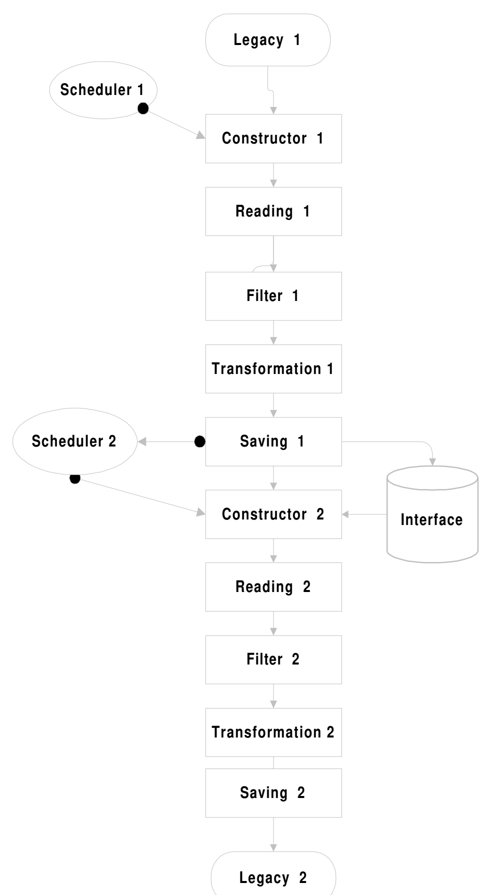

Figure 2 Base Architecture for a Data Middleware Product Line

group mainly interacts with the Product Engineering group. These relationships with customers suggest that the domain engineering is indispensable from the very beginning, and the product engineering is only necessary when there is a product already built.

The data middleware product line organization could be structured in three groups or areas we call clusters.

Cluster one is responsible for the domain engineering and it also includes the customers interacting with the domain engineering for building the first product. Cluster two deals with product engineering and also includes customers of subsequent products. Cluster three coincides with management.

In cluster one, either customers or domain engineers might have expertise in one or more similar implementations and platform-dependent solutions of a data middleware. If this were the case, the domain engineering is responsible for mining assets in these solutions, extracting their general design in a new generic one that will be the base architecture, and also some potentially reusable components. If nobody has a prior experience, everything needs to be developed from scratch. In any case, cluster one develops the first product of the line, even though this involves some tasks typically from the product engineering as system integration and testing.

The second cluster will carry out the implementation of subsequent products of the line. This group will receive the SPL scope, the base architecture, and a list of assets developed by the domain engineering. The data-middleware implementations should be developed reusing these assets as much as possible.

4.3. Base Architecture

Based on different real data middleware implementations and projects implemented [3], figure 2 shows a proposed base architecture for a software product line of data middleware products. This architecture contains key functions and components to be considered in the asset base and will be reused in future data middleware implementations.

The base architecture follows a pipes and filters pattern [5]. This pattern defines major processing components or filters that are the main assets of all data middleware products [3].

Note that this architecture has been developed for interchanging data between two legacy applications. For a data middleware development for more than two legacy applications, some new assets, including modifying the base architecture and adding new components and functions, should be considered.

Table 1 describes each component of the base architecture in figure 2.

Component	Description	Input	Output
Scheduler 1	This component will keep and control the trigger of the data middleware initial process.	Execution plan. Procedure manually executed File creation	Trigger executed Tracking
Constructor 1	Responsible for opening legacy one files and data structures. Basic validations are considered.	Files and fields mapping to access and read	Files open
Reading 1	This component is responsible for reading the information from legacy one files	Files open List of fields to read	Set of registers
Filter 1	Select and filter registers	Set of registers	Sub set of registers
Transformation 1	Transform and validate the subset of registers	Sub set of registers Transformation & Transformation rules	Sub set of registers transformed Subset of registers rejected
Saving 1	This component is responsible for saving the subset of registers transformed in a temporary legacy file	Subset of registers transformed Temporary legacy file open Pre conditions	Subset of registers transformed saved in a temporary legacy file Scheduler 2 triggered
Scheduler 2	This component will keep and control the trigger of the data-middleware secondary process.	Saving 1 triggered	Trigger executed
Constructor 2	Responsible for opening legacy two files and data structures. Basic validations are considered.	Legacy two files and fields mapping to access and read	Legacy two Files open
Reading 2	This component is responsible for reading the information from legacy two temporary file	Files open List of fields to read	Set of registers

Saving 2	This component is responsible for saving the subset of registers transformed in legacy two files	Set of registers Legacy two files mapped	Set of registers saved in legacy two files

Table 1. Base Architecture Component Description

4.4. Launching and Institutionalizing

As we discussed before, the implementation of product lines is not an easy task. Beyond that, launching SPL organization will need a clear interaction between clusters and strategic reevaluations. We suggest the use of the IDEAL [4][6][7][8] model for implementing, launching and institutionalizing because; with some generalizations this model allows process improvement and manage changes in an iterative manner.

The IDEAL model consists of five synergic steps as a process: Initiating, Diagnosing, Establishing, Acting and Learning. We describe each of these steps for the data middleware SPL.

<u>Initiating</u>. For the data middleware product line initiative, this step should involve building the business case as a well-defined document which will rule the product line scope, the market analysis and the way the funds will be acquired to meet the business objective. This period should take roughly 10-12 months. In this period only cluster 1 and a little cluster 3 are present.

<u>Diagnosing</u>. This step consists of the mining of a product candidate and its core assets in order to start building the product line and the first set of assets [10]. This mining activity is done based on the business case developed in the initiating step, the base architecture presented in section 4.3 and the company organization presented in section 4.2.

<u>Establishing</u>. In this step, a documented plan is elaborated and key people are trained. The plan considers all the tasks and responsibilities for each cluster, estimated efforts and cost for each task, and administration and control procedures for the management group. Once the plan is defined and validated, project kick-off is defined and check point meetings are scheduled and carried out periodically. All this information is formalized in a document and distributed to all stakeholders and people responsible for implementing the product line.

<u>Acting</u>. Given the objectives and the baseline defined in the business case, as well as the plan defined in the establishing step, this acting step involves the development of the products of the data middleware SPL. It also involves error detection and correction actions.

<u>Learning</u>. The learning step can identify any place where the product line effort does not match the business objectives or the organization context defined as part of the business case. This activity considers monitoring the acting and adjusting any of the prior documents, e.g. business case, plan, etc.

5. Conclusions

The development of a data middleware between two legacy applications is almost always a very complex project. Nowadays there are many legacy applications that need to interact. Developing one different product from scratch for every pair of legacy applications we need to communicate is not only a lot of work, but also a lot of unnecessary risk.

The software product line paradigm seems very attractive since it promises a very large scale reuse for a series of similar products as the data middleware. But developing reusable software involves many other qualities such as portability, visibility, abstraction, generality, configurability, among others.

The SPL is a new paradigm and new practice areas need to be considered to start the product line. We have presented a business case including some of the most important areas to be defined in order to apply the SPL paradigm in the development of a series of data middleware products. Practice areas presented in this BC involved the conformation of clusters for organizing the product line, the use of a proposed base architecture and a methodology for launching and institutionalizing the product line approach.

We have started building the SPL after building two ad hoc data middleware applications. We deduced the base architecture and followed the methodology described in section 4.4 to build a third data middleware product, actually the first one of the SPL adjusting it to the base architecture. We needed approximately 20 % less effort for building this product. We also build a second product in the SPL and we got even better results.

As part of our future work, we need to refine the base architecture in order to include the actual component implementations that could be reused in future products. Only with the base architecture we have obtained improvements in the software development process. We still need to define the base architecture at a lower level of abstraction to make this implementation reuse more straightforward.

6. References

[1] Bass, L.; Chastek, G.; Clements, P.; Northrop, L.; Smith, D.; & Withey - *2nd Product Line Practice Workshop Report* (CMU/SEI-98-TR-15)., J. Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1998.

[2] Bass, L.; Clements, P.; Cohen, S.; Northrop, L.; & Withey, J., *Product Line Practice Workshop Report* (CMU/SEI-97-TR-003). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1997

[3] Bastarrica, Cecilia ; Lopez, Marcelo. Software Product Line in Practice.Jornadas Chilenas de Computación 2001 – I Workshop de Ingeniería de Software. Universidad de Magallanes – Punta Arenas – November 2001.

[4] Bosch, J. "Organizing for Software Product Lines," Proceedings, 3rd International Workshop on Software Architectures for Products Families (IWSAPF-3). Las Palmas de Gran Canaria, Spain, March 15-17 2000. Heidelberg, Germany: Springer LNCS, 2000.

[5] Bushmann, Frank; Meunier, Regine; Rohnert, Hans; Sommerland, Peter; Stal, Michael – *A System of Patterns* – Wiley publishrer 1998 , pages 53 to 70

[6] Carnegie Mellon Software Engineering Institute (SEI) - *A Framework for Software Product Line Practice* – Version 3.0. available in http://www.sei.cmu.edu/plp/framework.html – 2000.

[7] Clements, Paul; Northrop, Linda - Software Product Lines (Practices and Patterns) – Addison Wesley Publications 2002

[8] Gary Chastek et al Software Engineering Institute - *Product Line Analysis: A Practical Introduction* - (CMU/SEI-2001-TR-001 and ESC-TR-2001-001) - June 2001

[9] Jacobson, I; Griss M; & Jonsson P, Software reuse: Architecture, Process and Organization for Business Success. New York, NY: Addison-Wesley, 1997.

[10] John Bergey, Liam O'Brien and Dennis Smith - *Mining Existing Assets for Software Product Line* - (CMU/SEI-200-TN-008) – May 2000