Structuring Software Cities – A Multidimensional Approach

Andreas Hess, Bernhard Humm¹, Markus Voss, Gregor Engels²

¹additionally Darmstadt University of Applied Sciences, ²additionally University of Paderborn

Abstract

Software cities alias application landscapes of large enterprises comprise tens or even hundreds of IT applications. Structuring software cities into domains is an important task of enterprise architects. The quality of the resulting domain model is crucial for the success of enterprise architecture management and an important tool for the governance of the development of an enterprise’s application landscape. This paper presents a novel method for constructing domain models based on business services, business objects, and business dimensions. The method has been validated in numerous industrial projects.

1. Introduction

The IT departments of large enterprises today run tens or even hundreds of IT applications. All applications, together with their interconnectivity, we call the application landscape of the enterprise. Application landscapes are sometimes also called software cities. If one compares the architecture of an individual application with the architecture of a building then the architecture of an application landscape is like the structure of a city. Like real cities, software cities need city planning to provide high quality support for the enterprise’s business processes and flexible adaptability to changes in the enterprise’s ecosystem.

Software city planning is the task of enterprise IT architects. As a discipline, enterprise architecture management addresses the structure of an application landscape (software city) as well as the management processes needed to govern the development of an application landscape from an as-is situation to a target structure.

Figure 1 shows the application landscape of a bank as an example. It shows the applications as components of the application landscape with their dependencies. It also shows a base structure (the coloured boxes) used to group the applications. In this paper these coloured zones of an application landscape are called domains. They define a high level structure for the application landscape – the domain model – that can be used as a tool in enterprise IT architecture management. In this paper, we present a method for constructing domain models for application landscapes. This method uses multiple dimensions (customers / markets, products, length of value chain, access channels to customers) to deduce the domains. This method has been proven in industrial practice.

Figure 1. Domain model and IT components of a bank

This work is part of the research effort Quasar Enterprise which has been conducted at sd&m Research and sd&m IT Consulting for more than three years. Quasar Enterprise distils the experience from enterprise architecture management and systems integration in numerous industrial projects with a total effort of more than 100 person years.

The paper is structured as follows: in Section 2, we introduce domain models. Section 3 defines the input for the method for constructing a domain model. This method is then presented in Section 4. Section 5 gives an example, followed by project experience (Section 6).
2. Application Landscapes and Domains

2.1 Terminology

Domains are the districts and suburbs of software cities. We use the term domain knowing that (like component) it is heavily used in different contexts with various meanings (e.g., domains in data modelling, domain specific languages, domains in component models like .NET, or business domains like banking). We chose to stick to it since it is widely established in the enterprise architects’ community [1], [2], [3].

The terms software city and application landscape are often used in publications on enterprise architecture as synonymous metaphors for the structure of an enterprise’s IT application portfolio. Examples are [4] and [5]. For the rest of this paper, we will use the term application landscape.

Domains represent clusters of services and managed information. They provide the top level structure of the application landscape. Domains may be refined by sub domains. In Figure 1, the domains are represented by the differently shaded areas of the application landscape. Domains group logical components which are bundles of services – the white boxes in Figure 1. The logical components are implemented using physical applications (commercial off-the-shelf or custom-built) that are integrated using an appropriate integration infrastructure, e.g., an enterprise service bus (ESB). The structure of the domains forms the domain model of the enterprise. Depending on the industry and the individual enterprise a domain model typically consists of approximately 10 - 30 top-level domains.

In this paper, we do not address groups of application landscapes, which would be a possible option for enterprise architecture management for multinational corporate groups.

We also focus on the top-level domain structure only. Domain / sub domain structures and component / subcomponent structures are possible. The presented method applies accordingly for different levels of granularity.

2.2 Why is a domain model important?

Domain models can be used by enterprise architects as a tool for defining the target application landscape at the enterprise level. Large Enterprises like Credit Suisse [1] or Deutsche Post [2] follow the strategy of a managed evolution of their application landscapes from an as-is situation to defined target. The managed evolution approach aims to define smaller IT projects with positive business cases and limited scope, so as to provide short-term business benefits while complying with a standard target.

The evolution is managed, because enterprise architects govern the development of the application landscape. In this way domain models are used for strategic planning as well as for the operational management of an enterprise’s IT project portfolio.

A domain model also helps to establish business ownership for applications and services as it links the business process architecture and the IT application architecture of an enterprise: domains represent sets of applications and databases that are managed as units for business reasons, instead of organizing them, e.g., by types of IT platforms.

Documenting application landscapes, another central part of enterprise architecture management, relies heavily on graphical and intuitive visualizations. Planning, deciding, and controlling with respect to measures on the application landscape all benefit from such visualizations, called software maps [6], [7].

Similar to the concept of maps used in cartography, a software map is composed of a base map and several layers. Figure 1 is an example of such a software map. Based on a defined domain structure, software maps can be used to visualize the delta between an as-is situation and the target situation defined by the domain model: The target domain model is used as a base map together with a mapping of the as-is application components on an individual layer.

Domains can also be helpful for more technical aspects of architecture management as they provide the starting point for the definition of rules for

- the design of components, e.g., components must not span domain boundaries
- the design of interfaces, e.g., loose coupling for interfaces used across domain boundaries
- the use of technologies for application integration, e.g. mandatory use of an enterprise service bus for the integration of components in different domains

3. Required input for constructing a domain model

The method for constructing a domain model deduces the domains from a set of input parameters that are part of the enterprise’s operating model.
3.1 Business services

Application landscapes are structured according to the business of an enterprise – not according to technology. Hence, the most important inputs for modelling domains are the business services of the enterprise. By business services we understand a top-level grouping of the main business activities that can be found in the value chain of an enterprise. Identifying business services is the task of business analysts. They can be derived from the top level business processes and the enterprise’s organization.

We distinguish between core business services and management & support services. Core business services directly facilitate the business strategy of an enterprise whereas support services only indirectly facilitate it. Business services can be structured hierarchically. We aim at about 5-10 top-level core business services. Figure 2 shows typical business services of a bank.

![Figure 2. Business services of a bank (examples)](image)

A bank comprises the following core business services:
- **Product development**, e.g., the development of new products and services
- **Marketing**, e.g., performing campaigns
- **Sales & advice**, e.g., advising retail customers on home loans
- **Operations**, e.g., selling stocks on the stock exchange (order execution)
- **Support service**, e.g., sending monthly account statements to customers

Management & support services are classic enterprise resource planning services (e.g., accounting, controlling, human resources), as well as risk management, compliance, treasury etc.

Since business services are top-level and coarse-grained, they are similar for different businesses. For example, an automotive manufacturer also has business partners, products, contracts, and orders.

3.2 Business objects

Business objects are the most relevant top-level information items of an enterprise. In our banking example we identify the following:
- **Business Partner**, e.g., a retail banking customer
- **Product**, e.g., savings, checking accounts, loans
- **Account**, e.g., the account used for booking of payments
- **Contract**, e.g., the result of opening a checking account
- **Order**, e.g., a bank transfer from one account to another

Top-level business objects are coarse-grained. We aim at about 5-10 business objects.

![Figure 3. Business objects](image)

3.3 Business dimensions

The operating model of an enterprise is typically shaped by the following question: Which **products** are sold to which **customers / markets** using an how long is the enterprise’s **value chain**?

A bank may answer this question in the following way. The portfolio comprises the following **products**:
- **Account-based products**, e.g., savings bank accounts
- **Loans**, e.g., home loans
- **Securities**, e.g., stocks, bonds
- **Payments**, e.g., bank transfers

Products are sold to the following **customers / markets**:
- **Retail banking customers**, i.e., private customers
- **Corporate customers**, i.e., companies

The following value-creating operations are / are not carried out by the enterprise:
- **Product development, marketing, sales, and support service** are completely performed within the enterprise.
• Payment transaction banking is not performed within the enterprise but is being outsourced to a partner corporation.
• Credit administration and securities settlement and custody are candidates for possible process outsourcing in the future. Both are subtasks of operations.

Often in this context, the term business value creation chain or shortly value chain is being used. The length of the value chain defines which value-creating operations are carried out within enterprise itself and which are carried out by partners. The more tasks are performed within the enterprise the longer the value chain. The more tasks are outsourced to partners the shorter the value chain. Modern enterprises tend to shorten the value chain by concentrating on their core business and by outsourcing specific tasks to specialists.

Categories like products, customers / markets or length of the value chain we call business dimensions. See Figure 4 for the banking example.

![Figure 4. Business dimensions of a bank](image)

The dimensions customers / markets, products and length of value chain are relevant to most businesses. Additionally, there may be business-specific dimensions. For a bank, the following customer access channels may be relevant:

- **Branch**, i.e. agencies where clerks directly service customers face-to-face
- **Self-service**, e.g. automatic teller machines
- **E-banking**, i.e. offering banking products via internet portals
- **Call centre**, where clerks service customers via telephone

Again, identifying the relevant business dimensions and their values is the task of a business analyst. We aim at about 3-5 business dimensions with about 2-5 values each. The values of the dimensions are directly linked to the business strategy of the enterprise. E.g., a direct bank’s strategy is to focus solely on the customer access channels e-banking and call centre.

Business services, business objects and business dimensions of the operating model form the input parameters for our method to deduce the domains. It is presented in the next chapter.

4. A method for constructing a domain model

Structuring application landscapes by domains is the task of enterprise IT architects. Finding a domain model of high quality is crucial for enterprise IT architecture management. It requires a high level of expertise, both in the business and in domain modelling. Constructing a domain model in practice is a highly incremental process. It is top-down from the business strategy as well as bottom-up from the existing application landscape. It requires a lot of communication and alignment since the final domains and their names must be understood and used by numerous stakeholders.

Insofar, there cannot be a method for constructing a domain model which can be applied mechanically and in an uninformed manner which nevertheless leads to a domain model of high quality. However, the experience from numerous domain modelling projects shows that there are common patterns, particularly in the top-down driven parts of the process – the bottom-up driven parts are usually specific to the application landscape under consideration. The following method for constructing domain models comprises those patterns in form of concrete steps.

The input for this method comes from business analysts as described in the last chapter:

- **Business services**, divided between core business services and management & support services
- **Business objects** and their associations with business services
- **Business dimensions** and their values according to the business strategy

Given this input, the method is as follows.

1. **Core business services**: Take the top-level core business services as initial domain candidates.
2. **Business dimensions**: Split domain candidates according to the characteristics of one or more business dimensions if their handling substantially...
differs from a business point of view. Iterate this step as long as the business strategy demands further differentiation.

3. **Business objects**: Consider the top-level business objects. In which of the domain candidates are they being written, i.e., generated, modified, or deleted? Make new domain candidates for all business objects that are being written in more than one of the existing domain candidates.

4. **Management & support services**: Make domain candidates for all management & support services. They usually follow the structure of modules that enterprise resource planning (ERP) software uses.

5. **Finalize**: The domain candidates become the final domains. Find meaningful domain names that are understood and accepted throughout the enterprise – if not already done during the iterations. Find a meaningful and intuitive graphical representation of the domain model. Check for completeness of the model by mapping the physical as-is applications to the domains.

5. **Example**

We demonstrate the method for constructing a domain model using the bank example.

**Step 1: Core business services**

In the first step, the top-level core business services form the initial domain candidates. See Figure 5.

<table>
<thead>
<tr>
<th>Product Development</th>
<th>Marketing</th>
<th>Sales &amp; Advice</th>
<th>Operations</th>
<th>Support Service</th>
</tr>
</thead>
</table>

Figure 5. Core business services

**Step 2: Business dimensions**

The second step is the most difficult step since it requires fundamental knowledge of the business strategy of the enterprise.

According to the business strategy of the bank, the business service **sales & advice** is to be performed differently along the business dimension **customers / markets**, e.g., for **retail banking customers** and for **corporate customers**. Therefore, the domain candidate **sales & advice** is split into two domains. The resulting domain candidates are named **retail banking** and **corporate banking**. See Figure 6.

![Figure 6. Splitting sales & advice according to customers / markets](image)

The business service **support service** differs according to the business dimension **customer access channels**. The resulting domain candidates are named **branch systems**, **self-service**, **e-banking**, and **call centre**. See Figure 7.

![Figure 7. Splitting support service according to customer access channels](image)

Additionally, the business service **operations** differs according to both business dimensions **value chain** and **product**. See Figure 8.

![Figure 8. Splitting operations according to product and length of value chain](image)

The business service **operations** differs substantially for the different products and therefore different horizontal domain candidates are introduced. However, **accounting & booking** can be treated identically for all products. Therefore, one vertical domain candidate is being introduced. The split of the **execution of payments** along the value chain is motivated by the
business strategy to outsource the payment transaction banking to a partner corporation. For similar reasons the value chains for loans and securities are each split into two domains.

**Step 3: Business objects**

In this step, new domain candidates are being introduced for business objects that are written (i.e., generated, modified, or deleted) in more than one of the existing domains. For example, business partners are being written in branch applications, e-banking and in call centre applications. So, the new domain candidates business partner management, contract management, and account management are introduced. See Figure 9.

---

**Figure 9. New domain candidates for business objects**

The business object product is being read in most domain candidates, including branch applications, self service, e-banking, and call centre. However, it is written in the domain candidate product development only. Therefore, both aspects – the business service product development and the management of the business object product – are assigned to the same domain candidate, which is simply named product. See Figure 10.

---

**Figure 10. Domain candidate product**

**Step 4: Management & support services**

The management and support services become domain candidates. See Figure 11.

---

**Figure 11. Domains for management & support services**

**Step 5: Finalize**

The domain candidates identified so far become the final domains. Meaningful names have been found during the iterations. Figure 12 shows the final domain model.

---

**Figure 12. Final domain model**

The final domain model shows customer-facing domains at the top of the diagram. The centre shows domains for the core business services, along the value chain. Then, domains managing enterprise resources are listed. At the bottom of the diagram, domains for management & support services are shown.

---

**6. Project experience**

sd&m has analyzed and / or constructed over 20 application landscapes of large corporations of different industry sectors. In some of those projects, the method for constructing the domain model has been applied explicitly, in others implicitly. We have analyzed 18 of the resulting domain models according to the usage of business services, business objects, and business dimensions.

The analysis demonstrates convincingly that business services, business objects, and business dimensions play an important role and, hence, give evidence of the validity of the method described. Table 1 gives an overview of sample domain names from nine application landscapes in the industry sectors automotive, banking, insurance, public, telecommunication, and tourism.
In his article “A methodology for service architectures”, Jones comes close to the methodology we describe [11]. He also starts out with business services. Since his article focuses on the identification of services, the succeeding refinement steps differ from our method for identifying domains.

Fundamental works on the separation of concerns and modularization like, e.g., [12] are, as a matter of course, the basis of our method. Particularly the works of Parnas (e.g., [13], [14]) have laid out the basics for designing components in the small. Our method can be seen as an application of Parnas’ principles in the large.

Our approach has similarities with different forms of organizational structures of enterprises, e.g. matrix organizations that combine functional and divisional structures where divisions are defined by products, markets or customer groups.

8. Conclusion

Application landscapes are extremely complex. One important tool of enterprise architecture management is structuring them into domains. Domains are being used for strategic planning as well as for the operational management of an enterprise’s IT project portfolio – the driver for the evolution of the application landscape. Because of this, a domain model of high quality is crucial.

In this paper, we have presented a method for constructing a domain model. The method takes into account business services, business objects and business dimensions like customer groups, product groups / markets, customer access channels and the length of the value chain. The main contribution of this paper is a concise description of the method in five steps which cannot be found in literature so far.

However, the method is based on well-known architecture principles and has already been used successfully in numerous large-scale industrial projects. Although the method is relatively straightforward, a high degree of business expertise and modelling experience is still required when constructing domain models of high quality.

9. References


