Communications of the Association for Information Systems

Volume 27 Article 27

9-2010

Domain Architectures as an Instrument to Refine Enterprise Architecture

Wiel A. G. Bruls
IBM Netherlands N.V., wiel_bruls@nl.ibm.com

M. van Steenbergen
Sogeti Netherlands B.V.

R. M. Foorthuis
UWV, Business Services

R. Bos
Department of Information and Computing Sciences, Utrecht University

S. Brinkkemper
Department of Information and Computing Sciences, Utrecht University

Communications of the Association for Information Systems

CAIS

Domain Architectures as an Instrument to Refine Enterprise Architecture

Wiel A. G. Bruls
IBM Netherlands N.V.
wiel_bruls@nl.ibm.com

M. van Steenbergen
Sogeti Netherlands B.V.

R. M. Foorthuis
UWV, Business Services

R. Bos
Department of Information and Computing Sciences, *Utrecht University*

S. Brinkkemper
Department of Information and Computing Sciences, Utrecht University

Abstract:

Enterprise architecture is concerned with the fundamental organization of the operating environment of an enterprise. The enterprise architecture is used to plan and control the construction of the systems that populate the operating environment. As the scope covered can be considerable in large enterprises, introducing domain architectures to partition and detail the enterprise architecture is a plausible approach. We formulate prescriptive criteria that consistent domain architectures must meet. By integrating the creation of domain architectures into an extended strategic alignment model we develop a theory that accounts for both the creation, scope-setting and detailing. Based on the creation viewpoint we derive a multi-level classification taxonomy. The primary differentiator is that between domains that are created from business usage viewpoints and those that are created from solution construction viewpoints. Four cases of domain architectures from actual practice are described that illustrate the variety encountered. Domain classifications in all cases conform to the theoretical model. The criteria, the developed theory and the cases have both academic relevance as well as significance for practitioners.

Keywords: domain architecture, alignment, enterprise architecture, case study, IT architecture, information theory, exploratory, cognitive

Volume 27, Article 27, pp. 517-540, September 2010

The manuscript was received 4/12/2009 and was with the authors 5 months for 2 revisions.

Communications of the Association for Information Systems

I. INTRODUCTION

Today's enterprises routinely need to produce and deliver complex mixes of products and/or services and adapt them to changing market needs quickly [Bennet and Bennet, 2004]. To do this efficiently enterprises need to develop and orchestrate a large number of operational processes and activities into a well-organized business and IT operating environment [Porter, 1985, 1996; Wernerfelt, 1984, 1995; Bharadwaj, 2000]. In today's large enterprises these processes and activities are supported by a large number of business and IT systems that populate the business and IT operating environment [Ross et al., 2006; Koning et al., 2006]. We refer to solutions as the conceptual abstraction of these systems. It is the structure of these solutions that determines the adaptability and alignment of the enterprise to its strategic objectives.

Enterprise architecture is the instrument that establishes the enterprise structure. It does so by conceptually modelling the business and IT solutions as an assembly of parts such as processes, functions and infrastructure, that work together in a coherent and well defined way [Ross et al., 2006; Harmon, 2005; van den Berg and van Steenbergen, 2006]. Based upon this structure the enterprise architecture formulates the principles and guidelines for construction of the solutions. The enterprise architecture is used to plan, govern and control the detailed architecting and engineering of individual solutions by solution architects and engineers.

The scope that the enterprise architecture needs to cover, particularly in large enterprises, is considerable. If the planning and control that the enterprise architecture seeks to exercise is to be provided at any level of detail then distributing the architectural work across a population of architects is required. Introduction of domain architectures as an additional type of artefact that limits enterprise architecture contents to a certain scope allows for this. Within the domain's limited scope, a separate (group of) architect(s) becomes responsible for refining the architecture. Where the enterprise architecture addresses the full scope of the business and IT operating environment and solution architectures address the scope of a single solution for a particular part of the business and IT operating environment, the scope of a domain architecture addresses a level in-between.

The refinement of architectural control that the introduction of domain architectures aims to achieve requires some form of partitioning and some form of detailing. The partitioning produces a reduced scope that allows distributing the work. The detailing produces a more fine-grained structure that allows increased control of engineering [Pulkkinen, 2005, 2006; Foorthuis and Brinkkemper, 2007; van den Berg and van Steenbergen, 2006].

Typical examples of scope settings [Pulkkinen, 2005, 2006] that are encountered in the literature include business types of scope-setting and IT type of scope-setting. Business scope-setting uses as viewpoint a particular business product or business function such as Loans and Mortgages, or Sales and Delivery. IT scope-setting uses as viewpoint a particular type of application or platform technology such as Database, or Network.

The scope of the research in this article is that of a domain architecture as an artefact [Hevner, 2004]. The objective is to develop a theory for creation and refinement of domain architectures, and to assess conformance to this theory based upon examples from practice. We include case descriptions that help us with clues for formulating these theories and assessing them. Our research is exploratory, cases match this research design.

Developing theory around an artefact at the point in time when it is introduced into a discipline has many useful outcomes for both practitioners and the research community. Careful design of the artefact that ensures completeness and consistency is a prerequisite for its effective usage in practice [Hevner, 2004]. Where we turn to practice by using cases to develop artefact theory, an assessment of how effective the artefact is when used in practice is outside the scope of this article. It requires more extensive field research to test predictions of effectiveness that can be derived from the artefact theory in this article. As is acknowledged by literature on IS theory-building [Gregor, 2006], this is an established sequence of research foci in IS research with clear benefits accrued for each type of research.

The approach to theory-building that we take combines elements from several theory-building paradigms [Gregor, 2006]. Firstly we formulate prescriptive criteria based upon a domain as a logical concept that ensures that domain architectures will be created consistently with respect to governance and contents. These criteria guide our theorizing about domain architectures and are useful for practitioners as well when creating domain architectures. Secondly we take an explanatory approach to understand why different types of scope-setting of domain architectures exist. We use clues from the included cases and from related literature, and use these clues to link the domain architecture creation process into the existing body of knowledge. From this linkage we derive an explanation that uses creation viewpoints associated with stakeholders as a central concept. Thirdly we take an analytical approach by arranging the domain architecture categories found in a taxonomy. Cases are then used again to assess conformance to this taxonomy, and to illustrate variations in refinement.

Our theory-building thus has a prescriptive element in the criteria that ensure consistency, an explanatory element in the different creation viewpoints that account for different categories, and an analytical element that summarizes results in a taxonomy. Aside from the academic value that is in the analysis and understanding, the value of this research towards the architect as a practitioner is a consistent definition of the artefact. This allows clear assignment of responsibilities, clear scope-setting, and clear communication to the community of engineers.

The remainder of this article is structured as follows. Section II defines what we mean by domain architectures by positioning them relative to partial enterprise architectures and solution architectures. Section III gives a description of the research method. It is followed by Section IV that derives the domain architecture creation. Section V describes the information collected in the cases. Next Section VI assesses the spread of domain architecture instances as encountered in the cases and draws conclusions about the applicability of the theoretical model. Section VII elaborates the significance of our results and the relationship between theory and cases. Section VIII summarizes our findings and includes suggestions for further research.

II. WHAT ARE DOMAIN ARCHITECTURES?

In this section we discuss what domain architectures are and the features that we seek to focus on in this article. We define domain architectures by differentiating between partial enterprise architectures, domain architectures, and solution architectures. All are created through some form of partitioning but at different levels of detail and with different scope-setting as depicted in Figure 1.

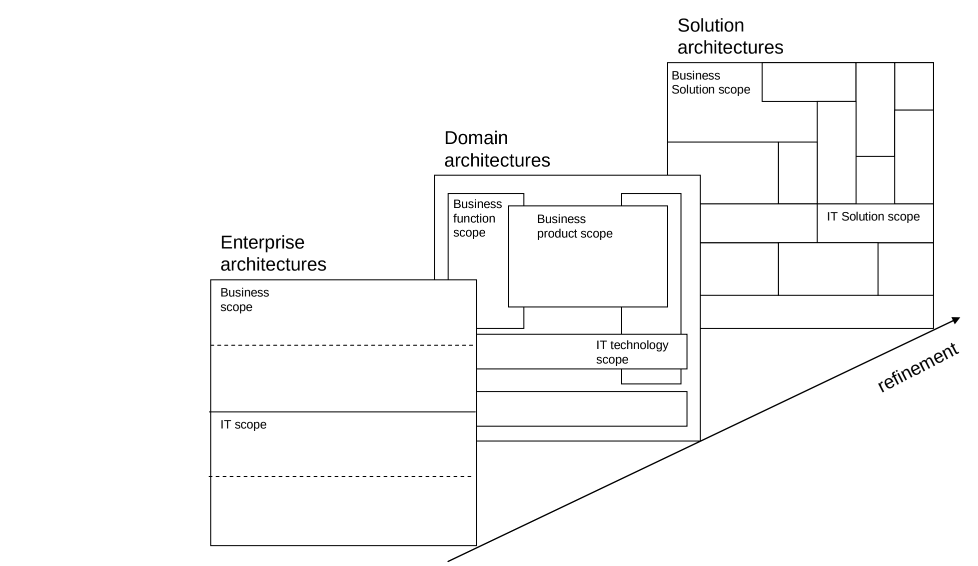

Figure 1. Architecture Partitioning

The figure positions partial enterprise architectures, domain architecture and solution architectures on different places on a refinement axis (running transversal through the picture from lower left to upper right). The scope covered by the architecture decreases with the axis, while the detail increases. At each layer a process of partitioning applies that is performed by selecting architectural contents within a scope set by a specific viewpoint [IEEE, 2000]. The figure shows at each level for each type of architecture examples with different scope-setting. What differentiates between the three types of architectures is the viewpoint used for partitioning, the degree of detailing (that increases with the axis) and the focus of the architecture (all solutions, a class of solutions or a particular solution).

Enterprise architectures. We define partial enterprise architectures as slices of the overall enterprise architecture that partition the full architecture into broad layers such as business architecture, information architecture and IT architecture. Note that our definition of enterprise architecture in this article follows the convention in most EA frameworks and encompasses all three; it is a broad definition of enterprise architecture—instead of a limited one that focuses on IS only. The type of contents and the focus towards engineering is at the same level as that of the enterprise architecture. The broad architectural layers that this results in are typically found in, for example, architectural frameworks that categorize the artefacts that the enterprise architecture consists of [Zachman, 1987; Sowa and Zachman, 1992; Schekkerman, 2006] and in architectural methods such as the method that is incorporated in The Open Group Architectural Framework (TOGAF) from OpenGroup that provide a systematic way for producing the enterprise architecture [TOGAF, 2009]. Differences exist in the exact number of layers and their terminology. The business layer, for example, may be split into process and organization [Iyer and Gottlieb, 2004], the information layer is sometimes referred to as *application layer,* and sometimes an explicit technology or IT infrastructure layer is added [Lankhorst et al., 2005]. Despite these differences, the viewpoint from which the partitioning occurs is derived from the transformation from business to IT, and the scope-setting from a focus on subject matter contents [Greefhorst et al., 2006].

Ross et al. [2006] use this partitioning as the base for the fundamental split of the operating environment of an enterprise in a business part and an IT part. Versteeg and Bouwman [2006] use it for the split into a business architecture (that they describe as the demand side of the business IT equation: it expresses what the business needs) and an IT architecture (the supply side of the business IT equation: what IT provides).

Although the layers that are produced could be labelled as domains (and in fact Iyer and Gottlieb [2004] do so), we do not consider this broad type of partitioning to produce domain architectures, as it lacks the element of detailing contents towards engineering. We consider these as partial enterprise architectures that make the development of the enterprise architecture itself manageable but at the same level of detail. Such initial partitioning allows development of the enterprise architecture to be performed by architectural disciplines with different focus. E.g., a business architect can consider the organization of the required business activities, and an IT architect can consider the construction of the different IT solutions.

Domain architectures. We position domain architectures at an intermediate level between enterprise architectures and solution architectures. Domain architectures are differentiated from enterprise architectures by the fact that they include an element of detailing and a reduced scope that are the result of a more fine-grained viewpoint and an increased focus on engineering of solutions. This definition is supported by various sources.

Pulkkinen [2005, 2006] describes a process for managing and refining architectural decisions at the enterprise, domain and system level producing increased detail at each level. In a number of selected cases from practice, the author finds that domain architectures are defined for a number of major business processes for which enterprise systems are planned, for a number of major information systems supporting core business processes, and for a number of functional areas of organizational activities. Van den Berg and van Steenbergen [2006] discriminate between enterprise architecture, domain architectures and project start architectures (at the solution level). Foorthuis and Brinkkemper [2007] define domain architectures as "architectures defined on the basis of one specific group of products, services, processes or functions." Coggins and Speigel [2007] introduce three levels for federated architectures of federal agencies: enterprise, segment and solution. Segment architectures correspond to architectures of a specific business line (a product group in terms of a commercial enterprise).

The TOGAF standard [TOGAF, 2009] describes as the primary mechanism for partitioning architectures the scope setting on solution subject matter. Two types of stakeholder viewpoints are identified that determine this scope: vertical viewpoints associated with business operational stakeholders that consider the solution landscape from a functional perspective and horizontal viewpoints that consider a certain piece of subject matter across the entire landscape. The level of detail increases with smaller scope settings.

Together these references provide a view—as depicted graphically in Figure 1 of domain architectures as architectures with a more fine-grained scope than enterprise architectures. The viewpoints from which they are created are quite narrow viewpoints around business entities such as business functions or products, or broader viewpoints around IT technology such as application or platform infrastructure. The cases included in this article that are presented after the theoretical model confirm this. In all of the four cases described, business focused domain architectures are found that use business functions and products as viewpoints, and IT-focused domain architectures are found that use application infrastructure as viewpoints such as Integration or Database, or platform infrastructure such as Network or Storage.

Solution architectures: We position solution architectures at the highest level of detail with as focus a single system that needs to be constructed to provide a business or IT solution usually through a project [Foorthuis and Brinkkemper, 2007; van den Berg and van Steenbergen, 2006; Coggins and Speigel, 2007; Pulkkinen, 2005, 2006]. The level of detail is highest and the scope smallest, the focus is fully on the subsequent engineering. We differentiate here also between business and IT solutions. For the first one the solution architecture develops the organization of the business processes and activities, and for the second one it develops the construction structure of the IT functionality.

Artefacts Used for Scope-Setting

Enterprise architectures contain a large variety of artefacts that are used in the various layers of the architecture. The scope of domain architectures needs to be set and the detailing needs to be described in terms of these artefacts. It is beyond the scope of this article to provide a full view of the transformations that are applied to the artefacts from the enterprise architecture to produce the artefacts from domain architectures and solution architectures. As a first step we will describe as part of the domain architecture creation theory the scope setting of the domain architectures. As enterprise architecture as a discipline bridges the gap between the strategies of an enterprise and the engineering of solutions, architecture methods include strategy-focused [Business Motivation Model, 2009] and engineering-focused artefacts as well [Youngs et al., 1999; Lankhorst et al., 2005]. As the core artefacts of the enterprise architecture, however, we consider the conceptual models that create the architectural structure. We will define scope setting and detailing in terms of the core architectural building blocks that are used to create these models [TOGAF, 2009]. In the domain architecture creation model we will provide the rationale for using these blocks as a base for scope setting; and we will identify how these blocks are selected and/or created for various types of domain architectures.

III. RESEARCH METHOD

The research method consists of a mixed method approach of combined theory-building and theory-testing using several inputs such as scanning of domain architecture literature, analytical literature review of relevant theories, and information extracted from case studies. The execution of our research according to these methods consisted of several iterations of evidence-finding, theory-building and theory-testing:

In the first step cases were selected and information collected. Case studies are considered useful when the topic is broad and complex, and when there is not a lot of theory available [Dul and Hak, 2008], as it applies here. The selection of the cases can be characterized as a combination of convenience sampling and purposive sampling [Teddlie and Yu, 2007]. Convenience sampling as the cases were selected from the practice of the authors. The authors had good knowledge of the architecture practices of the organization, but without having been involved in creation of the domain architectures. Purposive sampling as the organization had to be of different types, allowing our exploratory research to gain an impression of the variety. The individual organizations had to be in the process of creating or using the enterprise architecture, and the concept of domain architectures had to be used.

The sample that we obtained included one financial institution, one large governmental institution focused on information processing, one small governmental institution focused on intelligence, and one manufacturing company. These differ sufficiently in industry sector, size, product and customer scope, role and importance of IT and operating patterns, to allow us to explore differences in the use of domain architectures. This does not mean, however, that we pretend to have a fully representative set of cases that functions as a sample with which we can infer detailed statements about the full organizational population. Key informants of each organization checked the case descriptions.

In the second step, a first round of theory-building was performed based upon evidence from the cases and from information collected from domain architecture literature. The analysis approach used to extract information from the cases is the *cross-case synthesis*, in which the different cases are compared on individual features [Yin, 2003].

This analysis of the cases and the information obtained from the literature produced insight into the existence of technology-driven domain architectures and business-driven domain architectures. This fact was then used as a driver to construct the first version of the domain architecture creation theory. To this end an extended strategic alignment model from a number of selected areas from the body of knowledge was established and domain architecture viewpoints were integrated into this model in such way that the basic difference between the two categories observed from the cases could be explained.

In the third step additional detail was considered in the cases such as the split between primary business functions and secondary business functions, and the occurrence of application layers and infrastructure layers. Additional sources of structure were then identified (such as value nets and construction paradigms) to further extend the domain architecture creation theory and explain the identity of domain architectures within the two main categories.

In the fourth step the categorization was developed into a classification taxonomy. Conformance to this taxonomy was then assessed by classifying all domain architectures from the cases according to this taxonomy. Discussions between the authors on how to classify that in some instances resulted in considerable debate, were used to produce a set of classification guidelines that help to differentiate the domain architectures from the two main categories.

IV. DOMAIN ARCHITECTURE CREATION THEORY

In this section we derive the domain architecture creation theory that can explain the viewpoints from which domain architectures are created and the domain architecture's scope-setting and detailing. We start off with formulating a set of characteristics of a domain as a logical concept that are used as prescriptive criteria for the introduction of domain architectures as an artefact. Next we construct a simple model of an enterprise and how this is shaped by selecting from the existing body of knowledge from research fields such as strategic alignment, enterprise structure and competitive advantage. We refer to this model as the extended Strategic Alignment Model (eSAM). By integrating domain architecture creation into this model in such way that the prescriptive criteria are met, we are able to derive the viewpoints from which domain architectures are created. Based upon this we then develop a classification taxonomy. We also use the prescriptive criteria to select the architecture constituents on which scope setting and detailing is performed and we describe these mechanisms.

Characteristics of Well-Formed Domains

Merriam Webster [2009] defines a domain as "a territory over which dominion is exercised," "a sphere of knowledge, influence, or activity," or "a region distinctively marked by some physical feature." This definition combines *governance* and *contents* as two key features. Two sources of potential issues arise that are associated with these two features when creating domains. One is embedding domains in the enterprise governance processes and the other the partitioning process itself. When this is done incorrectly, the first may result in issues such as unclear ownership and lacking commitment of stakeholders. The second may result in issues such as disputes on scope and fragmentation of work amongst domain architects, and results that are difficult to use by the engineering community. In this section we establish the characteristics that well-formed domains must have to prevent such issues.

Domain governance: domains must be created in such way that they cover an area that matters to a stakeholder and that the domain's output is put to use. This results in the following governance characteristics:

- *Relevance*: to ensure that domains identified matter and have relevant contents, domains should be focused on the needs of a stakeholder
- *Ownership*: to ensure that clear responsibilities exist for the execution of a domain's obligations, domains should be controlled by an authority

For domain architectures in the context of an enterprise, this implies that they need to reflect the interests of key stakeholders such as business owners that represent the demand side of what the business is asking from solutions, or solution providers that represent the supply side of this equation.

Domain contents: domains must be created with clear scope and related contents that allows work in the domain to proceed in a well structured way and to produce well-structured results. This prevents issues such as debates on responsibilities, fragmented work processes and output, and inconsistent or conflicting combinations of instructions towards the community of engineers from different domains. This results in the following contents characteristics:

- *Boundedness*: to allow clear scope of responsibility, domains should have clear boundaries
- *Cohesiveness*: to allow work in a domain to proceed relatively independently and to produce well structured output, domains should have related contents with a focus that differentiates from other domains
- *Composability*: to allow combination of results into overall engineering guidance, domains should provide results that are compos able with that of others towards engineering

For domain architectures in the context of an enterprise this implies that the selection process through which the contents of the domain architecture is selected and the architectural substrate on which this occurs, need to create clear boundaries, cohesive contents and a clear focus. Results of different domain architectures should be formulated in such way that they can be composed. Note that although domain architectures require a focus that differentiates them from others, they are not intended to be disjoint by definition. Overlap is allowed and can even be sought for, to create tension between business stakeholders that are often quite narrowly focused on solutions that matter for them, and those stakeholders with a responsibility for corporate wide synergy.

In the theoretical model that we develop next we will use the required governance characteristics to account for the difference between business and technology viewpoints. They determine the place where the domain creation model is inserted into the eSAM. And we will use the required partitioning characteristics to assess the detailed categorizations and determine the enterprise architecture constituents that the scope of a domain can be set upon.

Creation Viewpoints

In this section we develop the theory part that explains the mechanism behind the different types of domain architecture creation viewpoints. To derive these viewpoints, we use an extended version of a strategic alignment model (eSAM) that integrates the development of enterprise solution structure into strategic alignment models [Chan and Reich, 2007]. The features of the eSAM should be able to account for the differences in domain architecture creation viewpoints as found in the cases, yet not be driven solely by the differences we seek to explain. This makes the relationship between the adopted eSAM and the domain architecture viewpoints derived from the cases a delicate one: they should not be causally connected. We solve this as described in the Section III—Research Method by using differences between domain architecture types from literature and the cases as high level clues that allow us to select relevant models and theories from the body of knowledge, and then integrate these selections into an eSAM. This anchors the explanation in the existing body of knowledge, and prevents an ad hoc explanation.

The feature that catches the eye most in a cross-case analysis of the domain architectures from the cases is the split in vertical business-aligned domain architectures with a scope set by a key business entity such as business product and functions, and horizontal cross-business domain architectures with a scope set by a key technology entity such as application and platform construction elements.

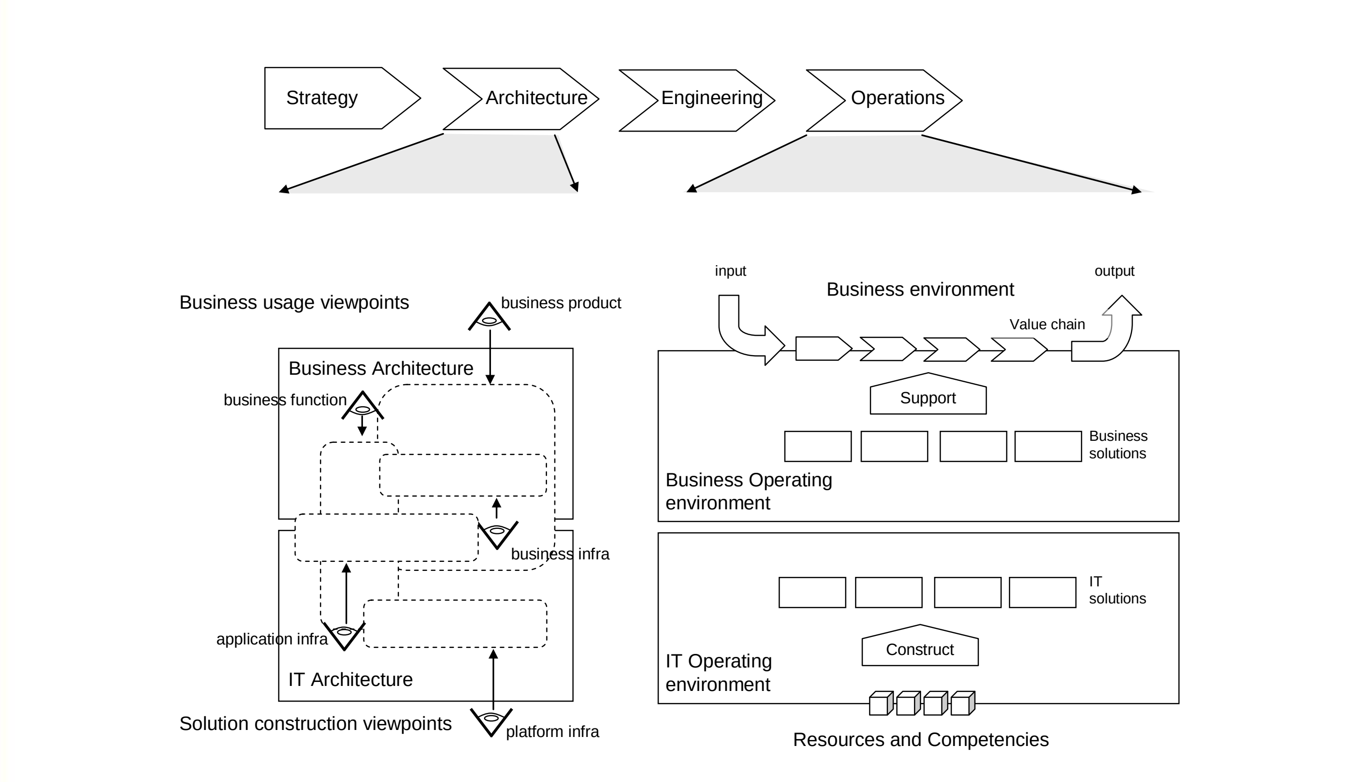

Figure 2. Extended Strategic Alignment Model

We turn to the raison d'être of architecture to interpret this division in business focused and technology focused domain architectures. Enterprise architecture, domain architectures and solution architectures are essential artefacts for shaping the solutions. They establish the structure of the solutions that populate the operating environment of an enterprise such that they can be *constructed* and then can be *used* as the enterprise requires them. *Usage* and *construction* are the key words here. By considering solutions from the *usage* perspective and from the *construction* perspective two basic viewpoints are created that reflect the interests of different stakeholders. Business owners are interested in the *usage* of those solutions that support the business they are responsible for (e.g., a product, a business function)—and less interested in other solutions. Technology owners on the other hand are interested in effective and efficient *construction* of solutions across all business usage out of shared application infrastructure such as Integration or Database, or platform infrastructure such as Network or Storage. Together the two perspectives *usage* and *construction* reflect a core theme in strategic alignment; that is to make sure that a solution does what the business owner needs it to do, but that its assembly reflects the synergy and adaptability that the enterprise as a whole will benefit from.

Guided by this essential difference, we now select from three areas of research to contribute to our eSAM: models of strategic alignment from business IT alignment research, operating models from enterprise structure research, and the resource-based view from research on competitive advantage. The resulting eSAM is presented in Figure 2. We next describe the features of the eSAM and how these features are taken from these selected areas from the literature.

The process of creating enterprise structure is depicted at the top of the figure and consists of four views: strategy, architecture, engineering and operations. The strategy view is concerned with direction setting, the architecture view is concerned with structure, the engineering view is concerned with construction, and the operations view is concerned with running of solutions [Ross et al., 2006; Harmon, 2005; van den Berg and van Steenbergen, 2006]. The architecture view is expanded in the lower left half of the figure and the operations view in the lower right half of the figure. The shaping of the enterprise through these views is based upon the strategic alignment model from research on business IT alignment [Henderson and Venkatraman, 1993]. The initial model consisted of a strategy view and an operations view, both split in a business and an IT part. Since domain architecture creation takes place at the level of the enterprise architecture and since the architecture then sets the guidelines for subsequent engineering, we have selected an extended version of this initial model that adds an architecture and an engineering view [Chan and Reich, 2007].

The model of the enterprise used in the operations view (in the right half of Figure 2) is derived from the literature on enterprise structure and how that produces competitive advantage. In this literature an enterprise is described as an organization that centres on a business production flow that interacts with the external business environment to process inputs, such as materials and information, into outputs such as products and services [Bennet and Bennet, 2004]. The source of competitive advantage is derived from the unique blend of resources and competencies that are used to construct the business and IT solutions and how these support the usage from the production flow [Porter, 1985, 1996; Wernerfelt, 1984, 1995; Bharadwaj; 2000]. The core elements from this description are the resource- and competency-based construction of solutions that is shown at the bottom of the Operations view, and the usage of solutions from a business production perspective that is shown at the top of the Operations view.

The resulting eSAM allows us now to adopt two different perspectives that match the usage perspective (looking down from the top of the figure) and the construction perspective (looking upward from the bottom of the figure). Viewpoints based upon these perspectives produce the two types of domain architectures in the Architecture view at the left in Figure 2 that we are looking for. More accurate: the viewpoints create domains by carving out parts from the business and IT operating model, for which the domain architectures create the solution structure, and this structure then sets the principles and guidelines for solution engineering. The usage and construction perspectives are layered perspectives that can consider the solution space from several levels as indicated visually in the figure.

Business usage domain architectures consider the solution space from a top level (a business product that oversees the full production flow) or from a view within the production flow (a business function is a limited part of the production flow). They select full solutions that support these major business operating entities with stakeholders such as business product owners and business production owners. This produces domain architectures such as “Loans and Pensions” domain architecture (centred on a business product) and “Sales and Marketing” domain architecture (centred on a business function). These domain architectures include contents across solution layers (including both business and IT solutions) and, therefore, have a vertical or at least a boxlike appearance when plotted under the business production flow.

Solution construction domain architectures consider the solution space from beneath at different levels of construction infrastructure: business infrastructure, application infrastructure and platform infrastructure. It opens the solutions up and selects common construction parts from the solutions, such as business assembly constituents, application assembly constituents or platform assembly constituents. Stakeholders are those enterprise roles that have a responsibility for construction infrastructure across the business, such as, for example, a CFO in case of payment infrastructure or a CIO/CTO in case of application or platform infrastructure. This produces domain architectures such as “Payment infrastructure” domain architecture (centred on a business infrastructure), “Integration” domain architecture (centred on an application infrastructure) and “Storage” domain architecture (centred on a platform infrastructure). Note that although in practice—as will become apparent from the cases—these domain architectures are mostly set up to cover specific IT construction technology, we do not limit these construction domains to technology by their definition. They can also extend to construction of business activities based upon competencies of human actors and thereby can support business solution that do not contain any technology at all. These domain architectures have a horizontal appearance as their contents are limited to a single layer of infrastructure, but include similar infrastructure parts across all solutions that support the production flow.

Inserting the domain architecture creation viewpoints in this way into the eSAM creates domain architectures that can meet the governance characteristics *relevance* and *ownership* almost inherently. Both types of domain architectures are created from perspectives that play a key role in business IT alignment. Business usage domain architectures created from the perspective of a major production entity match the interests of business stakeholders as they reflect external business potential associated with a product and its production (e.g., the revenue that sales of a product bring, the cross selling opportunity of a customer relationship business function, the just-in-time optimization of a supply chain business function, etc.). Solution construction domain architectures created from the perspective of a major solution construction entity match the interests of an infrastructure stakeholder, as they reflect the internal adaptive potential associated with a foundational production infrastructure (e.g., the integration capability of application middleware that can connect heterogeneous systems, the efficiency of a common storage solution that reduces cost, etc.). Domain architectures created from both these perspectives, when associated with major entities from business usage and solution construction, therefore, display the relevance and ownership characteristic.

Based on the principle of viewpoint categorization, we now construct a taxonomy (groupings based on the results of inductive, empirical analyses [Chan and Reich, 2005]) for domain architecture classification as depicted in Figure 3.

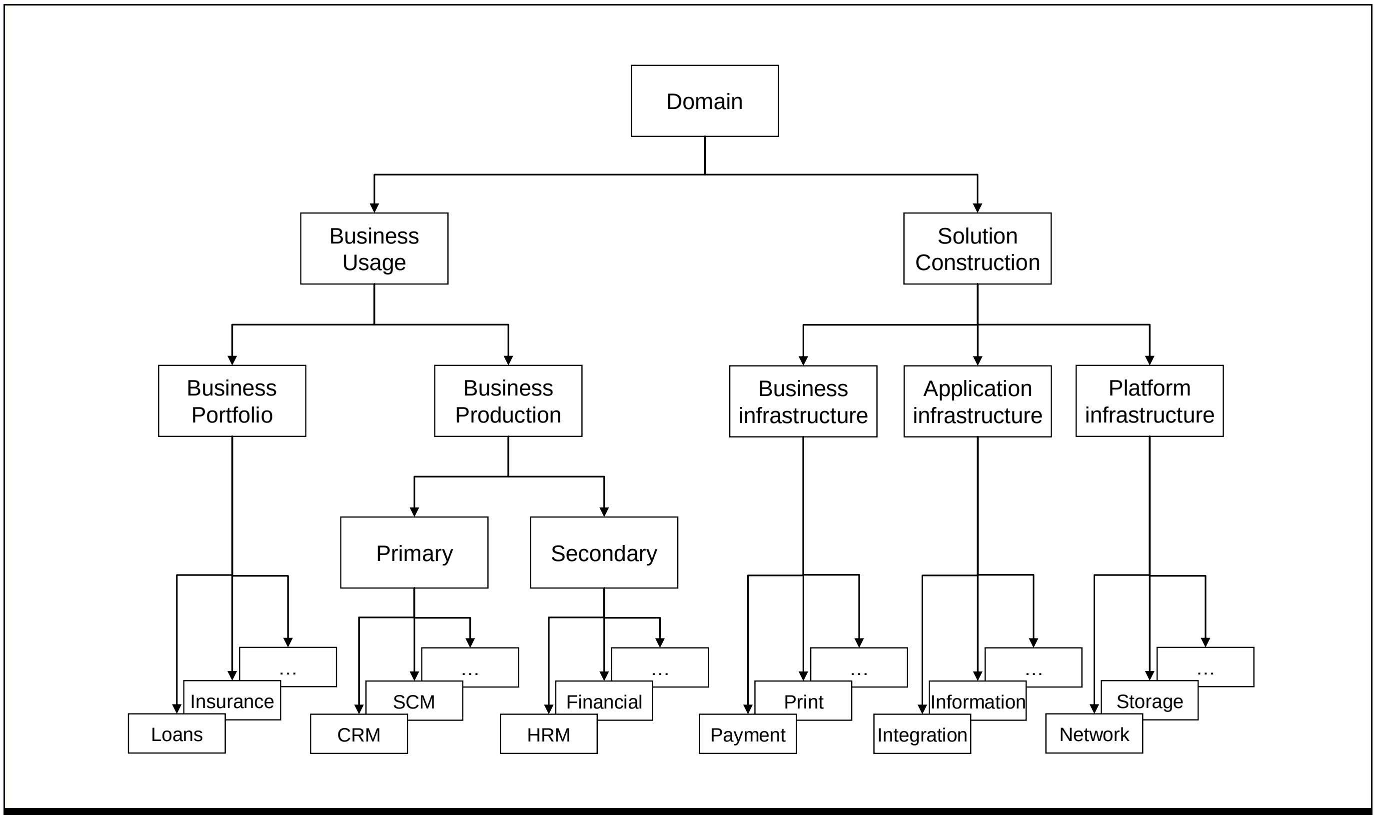

Figure 3. Domain Taxonomy

The figure shows three levels of domain architecture categorizations before the leaf level of domain architecture instances is reached. The primary split is that between business usage and solution construction viewpoints.

Business usage domain architectures are split into *Business Portfolio domain architectures* (with a grouping of externally available products and services from the business portfolio as domain architecture creation viewpoint) and *Business Production domain architectures* (with a grouping of internal business functions from the business production flow as domain architecture creation viewpoint). Examples of Business Portfolio domain architectures at the leaf level in a financial institution are a Loans domain architecture covering all solutions that support loans-related financial products, and an Insurance domain architecture covering all solutions that support insurance related financial products. Business Production domain architectures are further split into primary and secondary Production domain architectures, corresponding to the difference that Porter [1985] introduces between business functions that play a primary role in production value chain and those that play a support role. Primary domain architectures are created using core functions from the logistical flow through the value chain. This creates domain architectures at the leaf level such as Customer Relationship Management (CRM) and Supply Chain Management (SCM). The secondary functions are created around key supporting business entities such as Finance and Human Resource Management (HRM).

Solution construction domain architectures are split into three types of infrastructure domain architectures: business infrastructure, application infrastructure and platform infrastructure. This corresponds to the layered assembly of business and IT solutions in which common infrastructure can be identified at multiple layers [Lankhorst et al., 2005]. The domain architecture instances at the leaf level are split upon a simple conceptual model of the construction at each layer. At the business layer the conceptual model of the construction of the business includes a channel infrastructure, across which customer interactions are performed, and a payment infrastructure across which financial transactions are executed. This produces domain architectures such as Payment infrastructure and Channel infrastructure. At the application layer the conceptual model of the construction of the application includes a user interface part, a business logic part, an integration part and an information part. This produces domain architectures such as Presentation infrastructure, Service infrastructure, Integration infrastructure and Information infrastructure. At the platform layer the conceptual model of the construction of the platform includes the operating systems, the network and storage that host application construction elements. This produces domain architectures such as Network infrastructure and Storage infrastructure.

At each level of the taxonomy in this way the key conceptual entities from a dominant enterprise structure are used to create domain architecture categories and instances. These entities (e.g., the primary functions in the value chain, the conceptual parts that an application consists of) are used as viewpoints for creating domain architectures. They provide the focus that ensures that the contents related domain characteristics cohesiveness, boundedness and composability are met.

The proposed taxonomy is not exclusive; it can be extended as long as the consistency criteria are respected. For example, by adding nodes under business usage such as for business unit, or adding a tertiary production category under business production (for production control and planning). What counts is the principle behind the taxonomy: splits are created using as viewpoints key concepts from a structure that respect the consistency criteria.

Scope-Setting and Detailing

In this section we aim to establish the constituents from the enterprise architecture that are used as the base for setting the scope of domain architectures, and the process of detailing their contents. Our starting point are the contents characteristics that well-formed domain architectures must have.

Scope-setting is performed by selecting architectural contents within the domain architecture's creation viewpoints. We will now identify both the type of architectural contents on which the scope setting is performed as well as the type of mechanism through which the contents is selected.

Enterprise architecture contents consists of a large number of artefacts, with various notation techniques. Some of them are natural language based such as principles that are general rules and guidelines [Lindström, 2006] and such as capability models that state what an enterprise should be capable of to perform its strategies [Tulsky and Bagchi, 1998]. Some are ontology based such as taxonomies of patterns and business models [Osterwalder et al., 2005]. Some are model based such as solution architecture models, and high level engineering models [Dietz, 2006].

The partitioning criteria boundedness, composability and to a lesser degree the cohesiveness characteristic are most easily met if a *modelling* representation is selected as the core artefact on which scope setting will be performed. The reason is that models combine elements into a coherent scheme, using relationships between model elements. Splitting models by cutting ties between elements are orderly operations that can maintain the desired characteristics, if the split and the model are well chosen. Such splits are much more difficult on other artefacts (language based for example).

As the architectural artefact for scope-setting we select the high level architecture building blocks that are identified in architecture methods [TOGAF, 2004], that are sometimes referred to as solution areas [Schekkerman, 2006] or business components [Flaxer et al., 2005]. These architectural building blocks establish the structure for the engineering of the business and IT solutions. An example is a Customer Relationship Management block that after engineering will include processes and functions to handle the relationship with a customer. Thus these architectural building blocks are not actually used for building themselves nor do they map directly to solutions. They provide the envelopes in which solution construction can be performed using modelling artefacts such as business processes, business activities, application components, services, information flows, data stores, etc. Architecture building blocks at the level of the enterprise architecture and domain architectures are quite amorphous and may contain any number of these entities.

The contents of a domain architecture are now established as the selection of architecture building blocks that fall within the viewpoint from which the domain architecture is created. This mechanism ensures that the domain architectures inherently have the boundedness characteristic as blocks are included fully. The composability and cohesiveness characteristic requires the conceptual structures as base for partitioning as already described in the previous section. For example, the key concepts that constitute an application assembly model that produce domain architectures such as Presentation infrastructure, Service infrastructure, Integration infrastructure and Information infrastructure. Because of the underlying conceptual structure, solutions parts engineered under control of these domain architectures will blend into a seamless overall solution.

Domain architectures like enterprise architectures include more artefacts than just the architecture building blocks. Which architectural work products are included depends on the purpose of the domain architecture and their level of refinement. Domain architectures that focus on engineering will mostly include engineering-related work products such as modelling views and additional domain-specific principles. Domain architectures that also refine the business vision will include additional strategy-related work products.

The process of selecting architectural building blocks requires a different selection mechanism for business usage and for solution construction domain architectures. For business usage domain architectures, architectural building blocks need to be selected that together provide the structure for a full (set of) business and/or IT solutions (producing vertical domain architectures). This emphasizes the spatial aspect of the architecture modelling: architectural building blocks are close together in terms of working together. For solution construction domain there is a need to select architectural building blocks that play a similar role in solution construction across all business and IT solutions (producing horizontal domain architectures). This emphasizes the role aspect of the architecture modelling: architectural building blocks act in the same role but at different places.

This difference is similar to the different grouping mechanisms Jørgensen [2004] identifies in the domain of software engineering: composition reflects groups of elements that are connected through assembly while typing reflects groups of elements that have similar assembly properties but are not part of a single assembly. On an architectural level we propose instead the terms proximity-based selection and similarity-based selection for these mechanisms. Proximity-based selection serves as the selection mechanism that creates business usage domain architectures by selecting architectural building blocks that specify solution parts that will be assembled together and work together to provide the complete solutions these architectures define. Similarity-based selection serves as the selection mechanism that creates solution construction domain architectures by selecting architectural building blocks that specify solution parts that play the same role in construction of solutions. Where the first mechanism selects a complete set of blocks of different types per (set of) solutions, the second mechanism is focused on the commonality in construction and selects a set of the same type that provides infrastructure across solutions.

Guidelines for Classifying Domain Architectures

In this section we provide some examples of domain architecture classification and the complexities involved, and include some guidelines. As with any taxonomy it is not always trivial to determine how domain architectures need to be classified.

One question that arose when examining the cases in the comparative case analysis (see previous section on Research Method) was the difference between a generic implementation and a specific implementation. The synergy that horizontal solution construction domain architectures can easily establish for solutions across business operations is certainly an element of difference with respect to vertical business usage domain architectures. Although generic implementations for infrastructure are more common, in case of either business usage or solution construction domain architectures, the solutions can be implemented as generic solutions or a number of more specific ones. Consider, for example, an integration domain architecture that defines integration solutions. In some cases these solutions are implemented through a generic integration backbone, but in other cases a specific integration infrastructure solution is required that is optimized for the more demanding requirements of a specific business solution. And consider a customer relationship management domain architecture that defines CRM solutions for the various business products. This may be implemented through a single generic system that can serve the needs of the full set of product and customer segments in the enterprise. Therefore, the fact that the implementation of the solutions described in a domain architecture is performed in a generic way cannot be used for classifying a domain architecture as a solution construction domain architecture.

As another example of the complexity of classifying consider a parts warehouse from an automobile manufacturer. The solution that supports this may be included as a separate Parts management solution of a Supply Chain management solution. It is defined under control of the Integrated Supply Chain domain architecture that qualifies as a business usage domain architecture. Alternatively, the company may be pursuing a Just-in-Time delivery vision, and may have decided, therefore, to introduce for the Parts Inventory solution a separate Parts Management domain architecture. This domain architecture needs to create a consistent set of solutions that not only stores the parts as received from suppliers, but also handles their distribution across the business production line and local temporary storage, and further on to a self service portal for dealers. As it acts as infrastructure for the production applications along the production line, in that case it would classify as a business infrastructure solution construction domain architecture that is focused on business infrastructure. In an information intensive company, such as a Tax or Statistics agency that we will encounter as part of the cases, the equivalent may be an Information Management domain that supports solutions for storing and distributing information for use by all the core production processes.

The following table summarizes the differentiating characteristics of the two categories of domain architectures, and can be used as the base for classifying in business usage and solution construction domain architectures:

Table 1: Classification Guidelines

Aspect	Business usage domain architecture	Solution construction domain architecture
Scope	Support a selected *part of the full* business and IT operating environment	Support *(a large part of) the full* business and IT operating model
Viewpoint	Created from the *usage viewpoint* (business product or function)	Created from the *constructional viewpoint* (business infra, application infra, platform infra)
Functionality	Business functionality directly used in business operations	Infrastructure functionality indirectly used in business operations
Consisting of	Architecture building blocks that *work together* to provide a solution	Architecture building blocks that *play a similar role* across solutions
Selection	*proximity* based selection	*similarity* based selection
Stakeholder	Enterprise roles with *responsibility for a specific part* of the business	Enterprise roles with a *responsibility for construction infrastructure*

The table may be read in another way also: valid domain architectures are created by applying these classification criteria as normative statements.

V. CASE STUDIES

This section introduces four examples of domain architectures. The organizations that they are taken from include a financial institution, a manufacturing company, a national statistical agency and a governmental organization. The information collected includes information on the enterprise profile, the enterprise architecture contents, the domain architecture scope and type of creation viewpoints, and the domain architecture contents and its usage by practitioners. The information has been used in Section IV. Domain Architecture Creation Theory to understand the features that the theory needs to account for. And it will be used in Section VI. Evaluation to assess the extent to which the domain architecture classification taxonomy is complete and correct, and to assess how domain architectures position on the refinement axis between enterprise architecture and solution architectures (Figure 1).

Case A: A Government Organization

Enterprise profile: Company A is a government agency that is responsible for processing a diverse portfolio of tax regimes and social security supplements. Information systems in the past have been developed specifically for each. This has resulted in a number of production factories that are optimized for a specific product. Communication channels that are increasingly becoming electronic are adding a layer of channel function that is common across the portfolio.

Enterprise architecture: the IT department of the agency has developed the first version of the IT part of the enterprise architecture. The basic premise of the architecture is to break the product factory (silo) paradigm and introduce a service-based structure. The modelling identifies high level architectural blocks based upon a generic view of an application that centres on an enterprise service bus, and makes available application services across this bus.

Domain architectures: although the enterprise architecture only covers the IT part, both business and technical domain architectures have been identified as depicted in Figure 4.

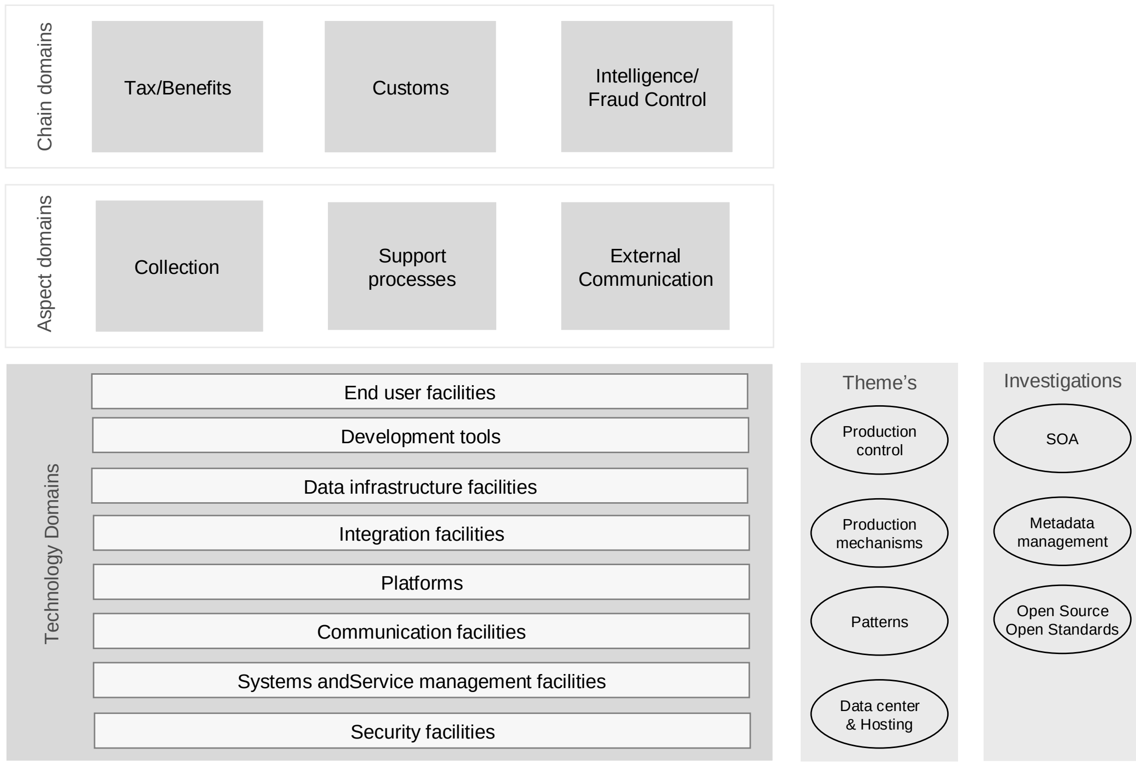

Figure 4. Domain Architectures—Case A

A limited number of business domain architectures have been created. They use business product and business functions as the viewpoints. Business product domain architectures cover business and IT systems that support operations on behalf of the major products: *tax/benefits*, *customs* and *intelligence/fraud control*. Business function domain architectures cover business and IT systems that support operations on behalf of major business functions: *money collection and disbursement* and *external communication* implement primary business production systems, and processes such as *enterprise resource planning* or *human capital management* implement business functions that support primary production processes.

Technical domain architectures use IT application assembly and platform support models as viewpoints. A number of topics that cannot be justified as full domain architectures but that still require an integrated approach at the enterprise level have been positioned as themes and investigations.

Contents and usage: the domain architectures include Market and Portfolio Developments, a Vision statement, a Solution view at three layers (business, application and infrastructure), and a Transition section. Engineering considerations are included to assess the technical feasibility. The product-oriented domains (tax, customs and intelligence/fraud control) reflect the current organizational boundaries. The business-function oriented domains reflect the core business purpose and strategic agenda: the collection domain is at the core of the business purpose of the organization (money collection and remittance), external communications reflect core elements of the strategic agenda (that aspires to change the product focus into a client centric and partner integration focus). At the infrastructure level the introduction of an integration domain follows the IT strategy to exit silo-based engineering in favor of a Service-oriented application model.

Domain architectures contribute to planning and governing. Actual development of information systems occurs in separate programs led by architects who take the enterprise architecture and domain architectures as guidance. The planning of enterprise IT development is performed at enterprise level, with the transition views of the domain architecture as input.

Case B: A Financial Institution

Enterprise profile: Company B is a large financial institution. It provides a wide range of financial services including savings, loans, payments, investments and insurances. In delivering these services the company also cooperates with third parties as intermediaries. Services are delivered through both the Internet and local offices.

Enterprise architecture: The enterprise architecture is divided into a business-oriented part and a technical part. The business-oriented part is structured into layers that cover interaction channels, sales and delivery, production of products and services and the internal operations. This enables company B to independently organize the systems of production and of distribution, to support multi-channel and multi-label communication and to separate the organization of the internal functions from the external functions. The technical part of the EA defines a common IT technical infrastructure, services architecture, process management, security and maintenance. The purpose of the enterprise architecture is the demarcation of IT programs and the application of rules and standards in domain architectures, project start architectures and technical implementation.

Domain architectures: domain architectures identified in the business part of the EA are a subdivision of the layering of the enterprise architecture as shown in Figure 5.

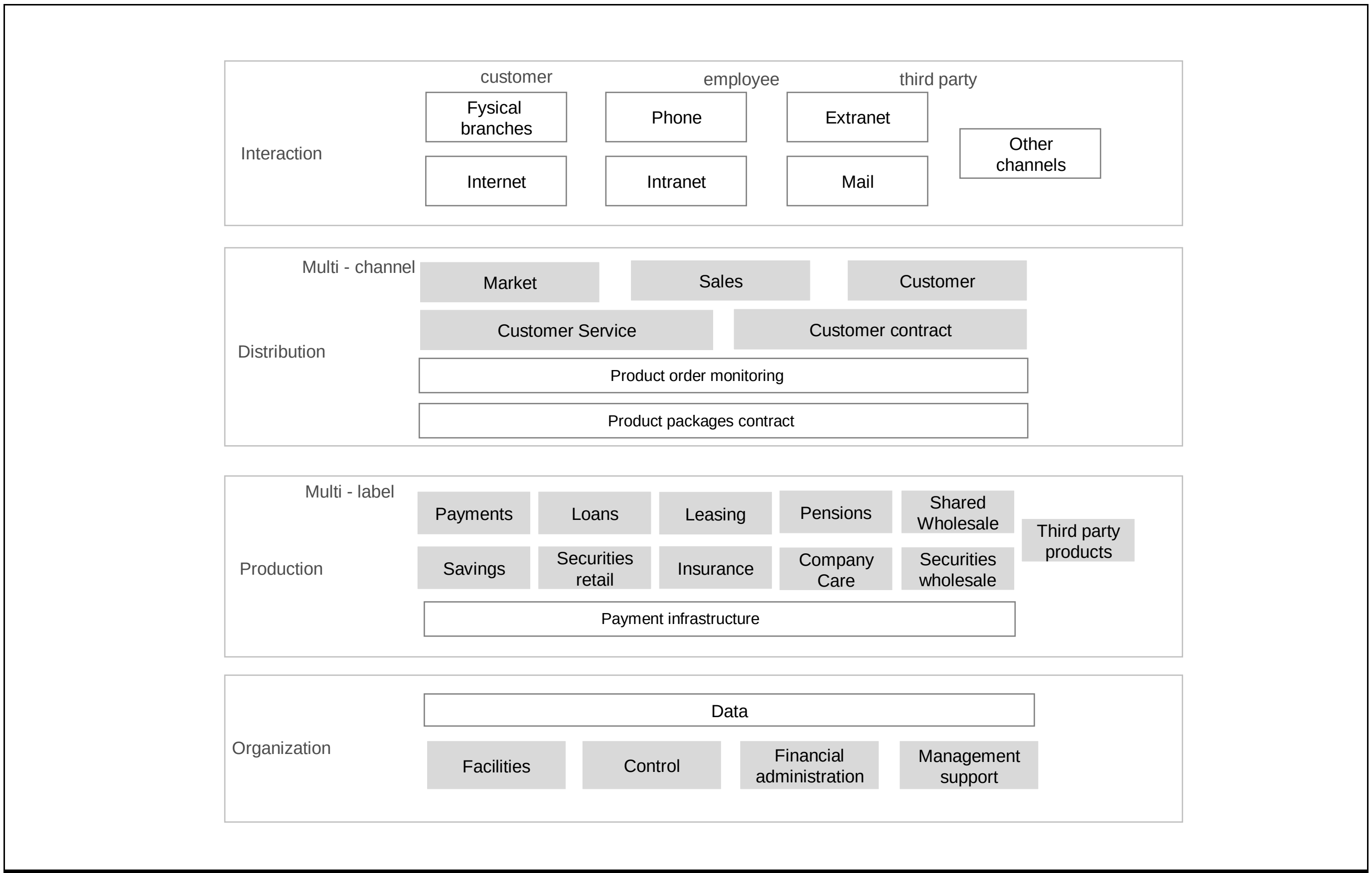

Figure 5. Domain Architectures—Case B

This creates quite a number of fine-grained domain architectures that come close to solution architectures. Even in this business part some domain architectures have a technical focus. In some cases a single domain architecture covers more than one domain. Business domain architectures are identified as follows:

- Within the interaction layer the fine-grained partitioning of the enterprise architecture into domain architectures has produced technology-oriented domain architectures per channel that are close to solution architectures and offer infrastructure services to business solutions.
- The distribution layer uses a function viewpoint to create business usage domain architectures: Market, Sales, Customer, Customer Services and Customer contract. Product order monitoring and product package contract can be considered construction domain architectures as they are included as functionality in each of the distribution layer domain architectures.

- The production layer uses a product viewpoint to create business usage domain architectures: payments, savings, loans, securities retail, leasing, insurance, company care, shared wholesale services and securities wholesale. Payment infrastructure can be considered as a construction domain architecture as it is used in multiple business solutions to communicate with the payment function.
- The organization layer includes a single solution construction domain architecture: data. It includes four domain architectures that are created from the viewpoint of business functions that support the primary production process: facilities, control, financial administration and management support.

The IT part of the EA is not formally divided into domain architectures apart from the partitioning that the enterprise architecture already contained.

Contents and usage: The domain architectures include a Requirements View (Business Requirements and Business models), a Solution View that consists of large number of models (such as Process, Function, Information, Components, Technology models), Engineering considerations (aimed at system realization), a Transition View (as is and to be situation and Migration plan) and Project Definitions.

The information management department is responsible for the formulation of domain architectures. Both the business representative for the domain architecture and the enterprise architecture team has to approve the domain architecture. The scope of the IT programs is precisely defined by the boundaries of one or more domain architectures. For example, a Sales IT program matches a Sales domain architecture. The realization of the domain architecture by way of an IT program is managed by the business representative, the program manager and the domain architect together. The division in domain architectures enables the company to better manage their change programs because of the alignment with the division in responsibility, which allows for clear decision making.

Case C: A National Statistics Agency

Enterprise Profile: Company C is a government organization employing over 2000 people. Its mission is to produce and publish statistical information about its country. As its input, throughput and output consist of information, the organization is very information and IT intensive by nature.

Enterprise Architecture: The EA features contents on various abstraction levels: the goals and context, the requirements to meet those goals, and the implementation-independent solutions, prescriptions and views to meet the identified requirements. The EA does not yet feature any implementation-dependent solutions, prescriptions and views. The main goals of the EA are reducing the costs of statistical processes, being able to create and redesign statistical products in an agile fashion, and improving the quality of statistics (e.g., coherence in publications, reproducibility of processes).

Domain architectures: A limited number of domain architectures are defined. All of them are business domain architectures as shown in Figure 6.

Figure 6. Domain Architectures—Case C

The domain architectures Data Collection and Data Dissemination are created around functions that support the initial and final process steps in the primary statistics production process. A major initiative to integrate the economic statistics is currently in progress. The domain architecture associated with this initiative integrates several related statistical processes that can be considered as the products of Company C. Other products exist, the most important being the social statistics; however, no explicit domain architectures have been defined here. The Data Service domain architecture and the Metadata Service domain architecture cover infrastructure services for storing and sharing statistical data and their descriptions to business solutions.

Contents and Usage: The specific goal of the domain architectures is designing the services identified in the EA in more detail and implementing them. They contain detailed solution views and an initial technology related set of engineering considerations. Some domain architectures have been acknowledged for years by the organization and have served as the key guiding mechanism for developing and consolidating applications. The application functionality covered by the Data Collection domain architecture has gone through several consolidation phases throughout the years, as several organizational units had begun to locally combine data collection tasks. Now, data collection is also centralized as a separate organizational unit delivering an enterprise-wide service. The functionality covered by the Data Dissemination domain architecture is provided by an automated Web publication application that makes a broad range of statistical data available to the public. At the moment, this domain is being redefined to include other services as well. The Service Centres introduced by the Data Service and the Metadata Service domain architecture bring goals and concepts that are relatively new to the organization. Consequently, their meaning and value for business users will have to become more tangible when actively in use.

Case D: A Manufacturing Company

Enterprise Profile: Company D is a multinational manufacturer. The company has plants in various parts of the world and employs about 23,000 people worldwide. The IT department consists of about 600 employees.

Enterprise architecture: Before developing the enterprise architecture, architecture was very much a collection of technological standards. There was no overall, comprehensive vision of business choices, processes and information systems, which inhibited the effectiveness of the architects. The purpose of the enterprise architecture is to provide a high level view of the situation to be based on the business strategy and to provide coherence between domain architectures. The enterprise architecture is divided into three parts: the business architecture, the information architecture and the technical architecture. Each of these three parts contains principles, policies and models.

Domain architectures: Domain architectures are created in each of the three parts of the enterprise architecture as shown in Figure 7.

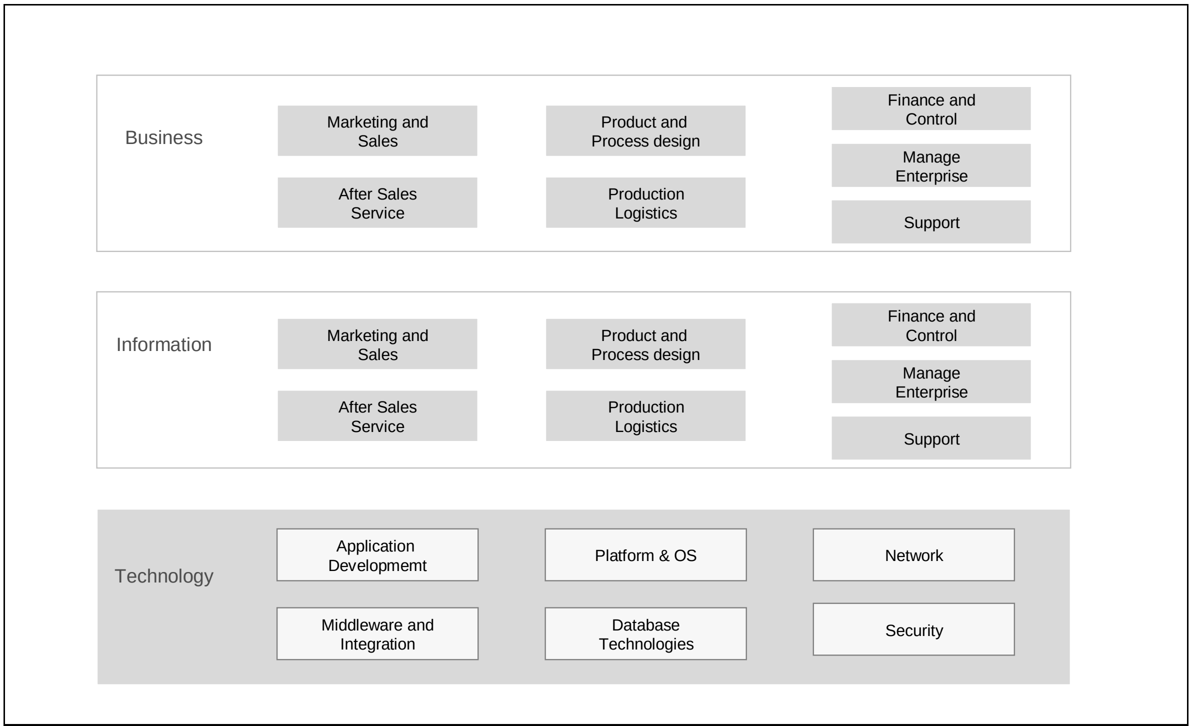

Figure 7. Domain Architectures—Case D

The domains in the business and information architectures are the same, as both are created from the same viewpoints. Within these viewpoints, the business architecture is concerned with business functions, while the information architecture deals with the supporting data and applications. All of the domain architectures are created from a functional viewpoint and support the primary production process, with the exception of the Manage enterprise and the Support domain architecture that are secondary functions. The boundaries of the domain architectures are not restricted by the current organizational boundaries. The technical domain architectures differ from the business and information domain architectures: they are demarcated on the basis of technical construction expertise.

Contents and usage: A business domain architecture describes the products, processes and organization of the business area. The information domain architecture describes the data and application of the business area. The technical domain architecture provides the guidelines on how to realize the technical dimension concerned. A domain architecture contains an analysis of the current situation, a design of the desired situation and how to migrate from the one to the other.

Work on the architecture has gone on for about three years. The development of domain architectures is the responsibility of the domain architects guided by the enterprise architects, and the domain architectures have to be approved by the enterprise architecture team. The information architecture domains are developed in close cooperation with the business lines. The division into domain architectures stimulates a closer collaboration between business lines and architects as it provides a clear means of expressing and realizing responsibilities.

VI. EVALUATION

In this section we assess the spread in domain architectures encountered in the cases and their conformance with respect to the theoretical model. Conformance with the theoretical model is tested by classifying domain architectures using the domain architecture taxonomy derived in Section IV. The spread is assessed by considering the level of refinement of the domain architectures based upon their scope and the type of artefacts they include.

Spread in Domain Architectures

Particularly for business domain architectures, a wide spectrum is found that varies between domain architectures that are large enough to be enterprise architectures in their own right, such as the product domain architectures found in case A, to domain architectures that are close to solution architectures as found in case B. Case D replicates the same domain architecture structure in both business and separate information domain architectures.

Technology is split out as a separate layer, with domain architectures identified in all four cases. Only in cases A and D are separate technical domain architectures created with assigned ownership. The overall organization of the technology domain architectures is quite the same, however, across the cases.

A higher level grouping is imposed on domain architectures, that matches the business to IT transformative layering as we introduced for partial enterprise architectures (Figure 1). The exception is the layering from case B that is solution functional layering (interaction, distribution, production) which does correspond to the fact that this is closer to a set of solution architectures.

Refinement level: (technical) construction domain architectures are positioned in all cases at the same level of partitioning and detailing between enterprise and solution architecture—decomposing based upon a conceptual platform and application model with a limited number of model elements. For the business usage domain architectures there is considerable variety in the partitioning and detail level, particularly in cases A and B. The very broad product domain architectures encountered in case A can be considered as “small” enterprise architectures in their own right that will require another domain refinement step. The parent organization can be considered as a “business of businesses.” The article by Coggins and Speigel [2007] describes similar findings and recommends a federated approach. In case B, domain architectures in the business view are created as a direct refinement of the enterprise architecture. They are at the level of programs for which almost a solution architecture can be produced. This implies that most decisions on the engineering structure have to be taken in the enterprise architecture itself.

In three out of four cases domain architecture contents are downstream (engineering) focused; they include models and guidelines to help construction and sketch an as-is and to-be situation to contribute to planning. Case A adds to that an upstream (strategy) focus by including vision and market development, confirming the fact that in this case business domain architectures are almost enterprise architecture in their own right.

The refinement level is summarized visually in Figure 8.

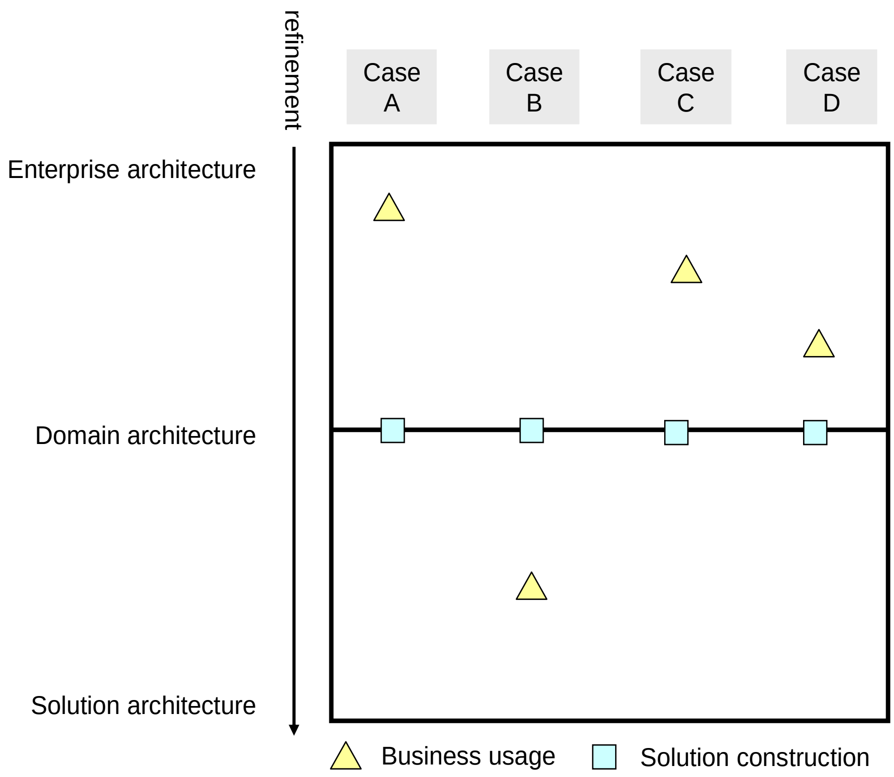

Figure 8. Refinement Level

In Figure 8, the domain architectures from the cases are plotted against the refinement axis from Figure 1 (shown at the left part of the figure). The solution construction domain architectures are at a fairly constant level of refinement, the business usage domain architecture scatter considerably.

Conformance with Theory

Domain architecture classification is well in line with the theoretical model developed, as is evidenced by the categorization in Table 2. This table assigns domain architectures from the cases to the values from the taxonomy summarized in Figure 3 (column 1 and column 2 from Table 2 use the values of Figure 3).

All domain architectures encountered in the cases can be classified according to the taxonomy. Business usage domain architectures use indeed viewpoints (business product and function) from the business operating model and can be explained by referring to the value chain model. Solution construction domain architectures occur at business, application and infrastructure level (the solution layering source of structure), and within these layers take their structure from the solution assembly paradigm.

Thus evidence from the cases overall complies well with the domain architecture creation model.

Table 2: Categorization of Domain Architectures									
		Company A		Company B		Company C		Company D	
Usage View	Business Product (Portfolio)	Tax/Benefits, Customs, Intelligences and Fraud Control	3	Payments, Savings, Loans, Securities retail, Leasing, Insurance, Pensions, Company care, Shared wholesale services, Securities wholesale, Third party products	11	Economic Statistics	1		
	Primary Business Function (Production)	Collection, External Communications	2	Market, Sales, Customer, Customer Services, Customer contract	4	Data Collection, Data Dissemination	2	Product and Process Design, Marketing and Sales, After Sales Service, Production and Logistics	4
	Secondary Business Function	Support Processing,	1	Facilities, Control, Financial administration, Management support	4			Finance and Control, Manage Enterprise, Support	3
Construction View	Business			Product order monitoring, Product packages contract, Payment infrastructure, Data	5	Metadata Services, Data Services	2		0
	Application	End user, Data infrastructure, Integration	3	Channels (Internet, Physical branches, Phone, Intranet, Extranet, Mail, other), Services architecture, Process management	3	Use of rule-based tools		Database technologies, Middleware and Integration	2
	Platform	Platforms, Communication	2	Technical infrastructure	1	Platforms, Security zones		Platform and OS, Network	2
	Others	Security, Systems and Service management	2	Security and maintenance	1			Application Development, Security	2

VII. DISCUSSION

This discussion focuses on two key aspects: first the outcome of our research and its significance, and second the delicate relationship between the developed theory and the cases that contribute to theory-building as well as theory-testing.

Outcome and Significance

Research presented in this article has focused on theory-building around domain architectures as an artefact: their creation, scope-setting and detailing. Artefact research is an established discipline [Gregor, 2006], some even consider the artefact as the core subject matter of IS research [Orlikowski and Iacono, 2001]. Without a clear shared view of an artefact practitioners are left to their private views, and research on the artefact's usage in practice and its effectiveness cannot be compared. Therefore, developing a theoretical view of an artefact at the point in time when it is introduced into a discipline is the primary research interest in the field of design science [Hevner et al., 2004] and benefits from case studies. It precedes research on the experimental validation of the effectiveness of the artefact in practice that can lead to predictive statements, which requires a different research design (such as surveys) and more extensive field research [Gregor, 2006].

Gregor [2006] in inquiring into the nature of IS research identifies five types of research (analytical, explanatory, predictive, explanatory and predictive, and prescriptive) and discusses the possible contribution to knowledge of each type of research. The theoretical elements included in this article match three of these types. Analytical research focuses on interpreting phenomena in a descriptive way, e.g., through classifying them in a taxonomy based upon attributes, as in the case of the domain architecture taxonomy. Explanatory research focuses on explaining why things are as they are, as in the linking of creation viewpoints and scope setting into the body of knowledge. Research for design and actions (prescriptive research) focuses on prescribing how things should be, as in the prescriptive criteria derived from a domain, as a concept that domain architectures need to comply to.

Possible contributions of prescriptive research are "the utility to a community, the novelty of the artefact and the persuasiveness of claims that it is effective." Domain architectures are artefacts that received little attention in academic research. The prescriptive criteria introduced in this article guide the analysis that anchors domain architecture creation viewpoints into the body of knowledge and ensures that the domain architectures proposed will be consistent, a prerequisite for effective usage in practice. In addition, the prescriptive criteria are useful for practitioners. They allow to assess the consistency of new categories of domain architectures that practitioners wish to create that are not covered by the taxonomy in this article.

Possible contributions of analytical research are "the production of meaningful categories, clear logic in classifying, and completeness in classification." Clear logic is used in establishing the classification principles that are behind the domain architecture taxonomy, and the categories produced are meaningful. The primary distinction in business usage and solution construction allows to balance the perspective of business owners that are focused narrowly on their interest (use of solutions in their domain) with that of construction owners that are focused on the broader synergy (common use of solution construction infrastructure). Examples of classifying are included in the article that demonstrate how the strategic importance of a choice for either by explaining the implications of such a choice. The logic for classifying is condensed in a number of guidelines that can be used by practitioners. The cases can be used by practitioners as well to illustrate the classification logic. Additional differentiators such as the value chain or construction paradigms, explain further differentiation of usage and construction categories. The taxonomy is not closed but is constrained by the fact that new categories that practitioners might want to introduce need to meet the prescriptive criteria.

Possible contributions of explanatory research are 'the production of new and interesting conjectures." The explanation of creation viewpoints and scope setting by integrating these in the body of knowledge from research on strategic alignment, resource based view, enterprise architecture and software engineering deliver such conjectures. Use of the resource-based view, for example, focuses attention on the resources that the operating environment of an enterprise is constructed from, and creates therewith a solid theoretical underpinning for the solution construction perspective as a source of key viewpoints for domain architecture creation.

Relationship between Cases and Theory

Cases and theory are presented together in this article for several purposes. First cases illustrate the use of domain architectures in practice and give through their details insight in the peculiarities that one may encounter and allows to gain an impression of the variety. With respect to the research, the cases contribute to both theory-building and theory-testing. The exact procedure is described in the Research Method section. The usage of cases for theory-building and theory-testing creates a delicate relationship: it could be taken to produce a theoretical bias as it uses the expected outcome as input as well.

There are two answers to this. First: in the science of the artificial [Simon, 1996], theory and data are indeed interlinked. Creating a design artefact is influenced by the practice that uses this artefact as it brings in the daily experience that then indeed underlies the theoretical laws. This implies that data from practice do shape laws to a certain extent. In addition, contingencies exist that can result in multiple valid description, which implies that there is no single theory of the truth, just an effective theory. This particularly applies to artefacts such as domain architectures that are instruments designed to achieve a result. They need to be effective in practice, by achieving the goal for which they have been designed. Second: the influence from the data of the cases is indirect only as the theory-building itself is done from the existing body of knowledge. The data from the cases are only used as initial clues that point to the appropriate areas of the body of knowledge to look for. Theory-building itself arises from linking the domain architecture creation process into the body of knowledge based upon these initial clues. In addition prescriptive criteria are established from domains as an abstract notion. This can be seen as an internal grounding of the theory (with external grounding in the data). Theory is thus constrained, on one hand, by the data (that need to be explained), and, on the other hand, by the concept. But the theory-building itself is largely done from the perspective of the existing body of knowledge.

The combined interplay between the body of knowledge, the domain as a concept and the cases illustrates the open dynamics of the process of introducing a new architectural artefact. The open-ended taxonomy together with the prescriptive criteria that apply when extending it, is the best example of this dynamic aspect. The taxonomy can be extended if new types of domain architectures not present in the current cases would show up, as long as they meet the prescriptive criteria as derived from the domain a concept. What is required for the new categories to meet these criteria are additional sources of structure like the value chain.

VIII. CONCLUSION

In understanding the use of domain architectures in the practice of enterprise architecture we have focused on their role as a possible refinement instrument that bridges the gap between the high level enterprise architecture and solution design. Based upon an extended strategic alignment model, we have developed a theoretical model for domain architecture creation; and using the principle of domain architecture creation viewpoints we have developed a classification taxonomy.

The four cases from practice included in this study illustrate the use of domain architectures as a refinement instrument. Domain architecture identities in the four cases conform well to the taxonomy derived.

The interplay between the evidence from the cases, the literature and the theoretical model, and the large spread in domain architectures in actual cases illustrates the dynamics of introducing a new artefact. It will take time to stabilize its usage.

Follow on research is required to focus on the effectiveness of using domain architectures, such as applying effectiveness tracing techniques [van Steenbergen, 2008]. Domain architectures are expected to contribute to the effectiveness of the primary goals of the enterprise architecture by (a) better alignment through the improved structure of solutions in the domain architecture, (b) better control and planning through improved identification of solution components and improved architecture control.

The research reported in this article has considered one instrument that plays a role in developing the enterprise structure as part of the process of business IT alignment. Research on the transformations and refinement mechanisms (such as proximity and similarity based grouping) through which enterprise architecture is produced out of strategy and refined towards engineering are at the core of the process of business IT alignment and require further work.

ACKNOWLEDGMENTS

The authors wish to acknowledge the work performed in the master thesis of Paul Speijers on various refinement types across architectural layers. We acknowledge guidance and suggestions by our editors and anonymous referees in various stages of submission.

REFERENCES

Editor's Note: The following reference list contains hyperlinks to World Wide Web pages. Readers who have the ability to access the Web directly from their word processor or are reading the paper on the Web, can gain direct access to these linked references. Readers are warned, however, that:

1. These links existed as of the date of publication but are not guaranteed to be working thereafter.
2. The contents of Web pages may change over time. Where version information is provided in the References, different versions may not contain the information or the conclusions referenced.

3. The author(s) of the Web pages, not AIS, is (are) responsible for the accuracy of their content.
4. The author(s) of this article, not AIS, is (are) responsible for the accuracy of the URL and version information.

Bennet, A. and D. Bennet (2004) *Organizational Survival in the New World: The Intelligent Complex Adaptive System*, Burlington, MA: Elsevier Science.

Bharadwaj, A.S. (2000) “A Resource-Based Perspective on Information Technology Capability and Firm Performance: An Empirical Investigation”, *MIS Quarterly* (24)1, pp. 169–196.

Business Motivation Model, http://www.businessrulesgroup.org/bmm.shtml (current Aug. 11, 2009).

Chan, Y.E. and B.H. Reich (2007) “IT Alignment: What Have We Learned?” *Journal of Information Technology* (22)4, pp. 297–315.

Coggins, C. and J. Speigel (2007) “The Methodology for Business Transformation v1.5: A Practical Approach to Segment Architecture”, *Journal of Enterprise Architecture* (3)2, pp. 45–55.

Dietz, J. (2006) *Enterprise Ontology—Theory and Methodology*, Berlin Heidelberg: Springer-Verlag.

Dul, J. and T. Hak (2008).*Case Study Methodology in Business Research*, Amsterdam: Elsevier.

Flaxer, D., A. Anil Nigam, and J. Vergo (2005) “Using Component Business Modeling to Facilitate Business Enterprise Architecture and Business Services at the US Department of Defense”, *Proceedings of the 2005 IEEE International Conference on e-Business Engineering.*

Foorthuis, R.M. and S. Brinkkemper (2007) “A Framework for Local Project Architecture in the Context of EA”, *Journal of Enterprise Architecture* (3)4, pp. 51–63.

Gregor, S. (2006) “The Nature of Theory in IS Research”, *MIS Quarterly* (30)3, pp. 611–642.

Greefhorst, D., H. Koning, and H. van Vliet (2006) “The Many Faces of Architectural Descriptions”, *Information Systems Frontier* (8)2, pp. 103–113.

Harmon, K. (2005) “The Systems’ Nature of Enterprise Architecture”, *IEEE International Conference on Systems, Man and Cybernetics, Vol 1,* pp. 78–85.

Henderson, J.C. and N. Venkatraman (1993) “Strategic Alignment: Leveraging Information Technology for Transforming Organizations”, *IBM Systems Journal* (32)1, pp. 4–16.

Hevner, A., et al. (2004) “Design Science in Information Systems Research”, *MIS Quarterly* (28)1, pp. 75–105.

IEEE, Std 1471-2000, “IEEE Recommended Practice for Architectural Description of Software-intensive Systems”, http://standards.ieee.org/reading/ieee/std_public/description/se/1471-2000_desc.html (current Aug. 11, 2009).

Iyer, B. and R. Gottlieb (2004) “The Four-Domain Architecture: An Approach to Support Enterprise Architecture Design”, *IBM Systems Journal* (43)3, pp. 578–597.

Jørgensen, K.A. (2004) “Modelling on Multiple Abstraction Levels”, *7th Workshop on Product Structuring, Product Platform Development,* Chalmers University of Technology.

Koning, H., R. Bos, and S. Brinkkemper (2006) “A Lightweight Method for the Modeling of Enterprise Architectures: Introduction and Empirical Validation”, http://www.cs.uu.nl/research/techreps/repo/CS-2006/2006-003.pdf (current Aug. 11, 2009).

Lankhorst, M., et al. (2005) *Enterprise Architecture at Work. Modelling, Communication and Analysis*. Berlin Heidelberg: Springer-Verlag.

Lindström, A. (2006) “On the Syntax and Semantics of Architectural Principles”, *Proceedings of the 39th Hawaii International Conference on System Sciences*.

Merriam-Webster, http://www.merriam-webster.com/dictionary/domain (current Aug. 11, 2009).

Orlikowski, W.J. and C.S. Iacono (2001) “Research Commentary: Desperately Seeking the ‘IT’ in IT Research—A Call to Theorizing the IT Artifact”, *Information Systems Research* (12)2, pp. 121–134.

Osterwalder, A., Y. Pigneur, and C.L. Tucci (2005) “Clarifying Business Models: Origins, Present and Future of the Concept”, *Communications of the Association for Information Systems* (16), pp. 1–25.

Porter, M.E. (1985) *Competitive Advantage,* New York: Free Press.

Porter, M.E. (1996) “What Is Strategy”, *Harvard Business Review* (32)1, pp. 10–20.

Pulkkinen, M. (2005) "EA Planning, Development and Management Process for Agile Enterprise Development", *Proceedings of the 38th Hawaii International Conference on System Sciences*.

Pulkkinen, M. (2006) "Systemic Management of Architectural Decisions in Enterprise Architecture Planning. Four Dimensions and Three Abstraction Levels", *Proceedings of the 39th Hawaii International Conference on System Sciences.*

Ross, J.W., P. Weill, and D. Robertson (2006) *Enterprise Architecture as Strategy*, Boston, MA: Harvard Business School Press.

Schekkerman, J. (2006) *How to Survive in the Jungle of Enterprise Architecture Frameworks: Creating or Choosing an Enterprise Architecture Framework, 3th edition,* Victoria, British Columbia: Trafford Publishing.

Simon, H. (1996) *The Sciences of the Artificial, 3th edition,* Cambridge, MA: MIT Press.

Sowa, J.F. and J.A. Zachman (1992) "Extending and Formalizing the Framework for Information Systems Architecture", *IBM Systems Journal* (31)3, pp. 590–616.

Teddlie, C. and F. Yu (2007) "Mixed Methods Sampling: A Typology with Examples", *Journal of Mixed Methods Research* (1)1, pp. 77–100.

TOGAF, http://www.opengroup.org/togaf/ (current April 5, 2009).

Tulsky, W.A. and S. Bagchi (1998) "Strategic Capability Networks", US Patent 6249768 filed on 1998-10-29, http://www.patentstorm.us/patents/6249768-claims.html (current Aug. 11, 2009)..

Van den Berg, M. and M. van Steenbergen (2006) *Building an Enterprise Architecture Practice*, Dordrecht: Springer.

Van Steenbergen, M., R. Bos, and S. Brinkkemper (2008) "The Application of a Prescriptive and a Constructive Perspective to Design-Science Research", *Proceedings of the 3rd International Conference on Design Science Research in Information Systems and Technology,* pp. 191–211.

Versteeg, G. and H. Bouwman (2006) "Business Architecture: A New Paradigm to Relate Business Strategy to ICT", *Information Systems Frontier* (8)2, pp. 91–102.

Wernerfelt, B. (1984) "The Resource-Based View of the Firm", *Strategic Management Journal* (5)2, pp. 171–180.

Wernerfelt, B. (1995) "The Resource-Based View of the Firm: Ten Years After", *Strategic Management Journal* (16)3, pp. 171–174.

Yin, R.K. (2003) *Case Study Research: Design and Methods*, *3th edition,* London: Sage Publications.

Youngs, R., et al. (1999) "A Standard for Architecture Description", *IBM Systems Journal* (38)1, pp. 32–50.

Zachman, J.A. (1987) "A Framework for Information Systems Architecture", *IBM Systems Journal* (26)3, pp. 276–292.

ABOUT THE AUTHORS

Wiel A. G. Bruls is an Enterprise architect in IBM's global services business in the Netherlands. He received a Ph.D. in Physics from the University of Utrecht in 1984 after which he joined IBM and worked as lead architect on development of commercially available IT and business systems. In 1996 he joined IBM services where today he consults with clients on enterprise architecture. Dr. Bruls works as a guest researcher at the University of Utrecht in the field of enterprise architecture. His research interests include the use of enterprise architecture in business IT alignment, strategic change and associated major business IT transformations.

Marlies van Steenbergen is principal consultant of enterprise architecture at Sogeti Netherlands with over ten years experience as an enterprise architect. She obtained her master in linguistics at Groningen University and in Computer Science at Delft University. She now works as an advisor and a coach in the field of enterprise architecture. Marlies van Steenbergen is one of the founders of DYA®, an approach to developing and improving an effective architectural practice, and co-author of the books Dynamic Enterprise Architecture, how to make it work and Building an enterprise architecture practice. She is currently working on her Ph. D. thesis on the effectiveness of enterprise architecture practices.

Ralph M. Foorthuis works as an analyst and architect at UWV, Business Services. He studied at the University of Amsterdam, where he obtained his master in both Communication Science and Informatics. He is currently conducting his Ph.D. research on projects that have to comply with an enterprise architecture. Research topics include best practices and models for projects conforming to EA, frameworks for project architecture and testing projects on compliance with EA.

Rik Bos is assistant professor at the Department of Information and Computing Sciences, Faculty of Science, Utrecht University. He obtained a Ph.D. in Mathematics in 1984, also from Utrecht University and then switched to Computing Sciences at the Technical University of Eindhoven. He has worked for fifteen years in several software companies in the field of software and enterprise architecture. His research interests include enterprise Architecture, Software modelling and patterns.

Sjaak Brinkkemper is professor of Organisation and Information at the Institute of Information and Computing Sciences of Utrecht University in the Netherlands. Before this he was a consultant at the Vanenburg Group and a Chief Architect at Baan. Before Baan he held academic positions at the University of Twente and the University of Nijmegen, both in the Netherlands. He holds a B.Sc. of the University of Amsterdam, an M.Sc. and a Ph.D. in Computer Science of the University of Nijmegen. He has published seven books and more than a hundred papers on his research interests: software product development, information systems methodology, meta-modelling, and method engineering. He is a member of the ACM, of the Computer Society of the IEEE, and of the Netherlands Society for Informatics.

Communications of the Association for Information Systems

ISSN: 1529-3181