# Enterprise Cartography: From Theory to Practice

Pedro Sousa[1,2], José Tribolet[1,3], and Sérgio Guerreiro[1,3]

[1] Instituto Superior Técnico, Av. Rovisco Pais, 1, 1049-001 Lisboa, Portugal
[2] Link Consulting SA, Av. Duque de Ávila 23, 1000 Lisboa, Portugal
[3] INESC-ID, Rua Alves Redol 9, 1000-029 Lisboa, Portugal
pedro.manuel.sousa@tecnico.ulisboa.pt
jose.tribolet@tecnico.ulisboa.pt
sergio.guerreiro@tecnico.ulisboa.pt

**Abstract.** We refer to Enterprise Cartography (*EC*) as the process of representing an Enterprise observed directly from reality. It differentiates from Enterprise Architecture (*EA*) because it focuses on producing representations based on observations, and therefore, does not include the purposeful design, as one expects in *EA*. To abstract and represent the enterprise, one requires a full set of familiar concepts from the *EA* discipline, such a model, views, viewpoints, representation rules, *etc.* However, to abstract and represent the enterprise reality, one also needs principles and instruments to deal with the continuous and fast transformations going on in the enterprise. Without them, enterprises will likely change faster than one can represent them, and representations become obsolete long before they are completed. We have been working on this issue over a decade and have come up with a set principles and instruments to enable effective *EC* in continuously changing enterprises. We have applied and improved our methods in several enterprises of different sizes, sectors, and countries, and have had our share of successes, failures, and lessons learned.

**Keywords:** Enterprise Cartography, Enterprise Architecture, Organisation self-awareness

## 1 Introduction

According to Merriam-Webster dictionary[47], Cartography is *the science or art of making maps*. It is an old and established discipline of producing representations of an object from its observation and measurement. Traditionally, these maps require not only geographic or spatial referential, but also scientific, technical and even aesthetic[51] ones. Unlike traditional maps, enterprise representations do not have necessarily a geographic or spatial referential. Nevertheless, since the definition of *Cartography* matches our concept, we adopt the term *Enterprise Cartography*, hereafter designed as *EC*.

Simply put, *EC* is the process of abstracting, collecting, structuring and representing architectural artifacts and their relations from the observation of enterprise reality.

In this chapter, the expression "enterprise reality" refers to the present state of the enterprise. Traditionally, the observation of this reality is a subjective perception of an observer, and consequently it is probable that there are different actors that perceive different realities of the same enterprise. However, as we propose and will become clear along the text, the perception of the present state of the enterprise (the reality) is sustained on relevant facts captured in logs and represented through artefacts based on previously defined and agreed upon models.

We refer to "architectural artifacts" as the enterprise's observable elements whose interdependencies and intra-dependencies express the architecture of the enterprise. Naturally, different institutions may consider various sets of artifacts as part of their architecture.

To abstract, structure and represent the architecture related artifacts, one needs the whole set of knowledge and concepts implied in architecture visualization and representation. For example, the concepts of semiotic triangle[23, 7], the model of architecture description presented in ISO IEEE 1471[16] and a symbolic notation, such as the one used in ArchiMate[25], are concepts necessary to the production of architectural maps.

Traditionally, these architectural maps can present the architecture at different points in time, namely at present (*AS-IS*) and at a given time in the future (*TO-BE*). Within the *TO-BE*, the architecture that is emerging as the result of ongoing transformation initiatives is of particular importance for all sorts of planning purposes, from the strategic macro levels, to the full synchronization of multi-project initiatives at tactical and operational levels.

Ongoing transformation initiatives and their plans are an essential part of enterprise reality because they define the near future *TO-BE* of the enterprise if no further decisions are taken that might have an impact on them. This is a fundamental concept that we call *emerging AS-IS*, which we define as the state of the enterprise after successful completion of ongoing transformations.

Given that the transformations underway are in fact an endless succession of transformations that begin and end, the concept of *Emerging AS-IS* is not only an essential capacity for the planning of new transformations, but also for the monitoring of the enactments of ongoing transformations. When driving a car the faster the car is moving, the farther ahead one should focus our eyes to match the longer time and distance context within which we need to steer it. Likewise, when steering an enterprise, the faster the enterprise is changing, the more important is to know the *emerging AS-IS*, as time flows.

Continuing with the car metaphor, if one knows the car situational reality (position, speed, and direction), then the car future's position, speed, and direction can be estimated unless some unforeseen action (*e.g.* break or turn) is taken. Much like the car's speed and direction, ongoing transformation initiatives are, in fact, the core elements of enterprise inertia, since they pre-condition the future of the enterprise (the *TO-BE*), unless some decisions are taken meanwhile to change such core inertial elements. Since plans of ongoing transformation ini-

tiatives are observable artifacts, by observing in real time the enactment of the enterprise reality, one can predict the expected future enterprise states.

*EC* is a purely descriptive perspective, since it does not explicitly incorporate the purposeful design of the new enterprise artifacts, as one expects in *EA*. Such a difference is also evident in the definitions of the *architect* and *cartographer* roles. According to the IEEE Standard Glossary of Software Engineering Terminology[17], the term *architect* is defined as *"The person, team, or organisation responsible for designing systems architecture"*, and in the Merriam-Webster dictionary[47], the term *cartographer* is defined as the *person who makes maps*. So, an *architect* is essentially a person that designs and shapes intended changes to the architecture of the present enterprise reality, while a *cartographer* is essentially a person that aims at representing reality as it happens, including such changes as they are occurring.

Cartographic change as it is happening is the most challenging aspect in *EC* since it implies doing representations of evolving objects. The real challenge comes when the practices of abstracting and representing the evolving enterprise occur at a rhythm that is slower than that the actual rhythm of enterprise changes.

Therefore, to abstract and represent enterprise reality, one also needs principles and instruments to deal with the continuous and fast transformation of the enterprise. Without them, enterprises will likely change faster than one can represent them, and representations become obsolete long before they are completed. This is, in fact, the application of the famous Nyquist's sampling theorem[53] applied to Enterpises according to General Systems Theory[22, 55]: the sampling rate of sensing events must be high enough to follow the underlying dynamics of changes occurring in the underlying physical environment.

In this chapter, we start presenting the key motivations for the practice of *EC* in real enterprises. First, we argue that accurate maps of the enterprise are a key asset not only to support day-by-day operations but also to support the planning of transformation initiatives. Secondly, we argue that keeping such maps continuously updated is easier when the roles of architectural design are kept apart from the roles of cartographic representation, thus enabling the separation of concerns between Architecture and Cartography.

We now present an overview of similar approaches to cartography existing in different domains, from infrastructure to business processes. Next, we introduce the set of definitions and principles that shape our approach to *EC*. After that, we present some lessons and conclusions from our experience in real enterprises, and finally, we present our conclusions on the usage of *EC*.

## 2 Motivation for Enterprise Cartography

In general, simplifying and streamlining enterprises entails not only new processes and systems, but also greater integration with existing ones. More and more integrated processes and systems result not only in an increase in enterprise complexity but also in a greater need to know the enterprise current and

emerging processes and systems. Unfortunately, these two factors feed each other in a positive feedback loop: the more complex the enterprise is, the more one needs to know, the harder it gets to know it. Such positive feedback significantly contributes to the increase in costs, risks and execution time for transformations initiatives, as well as the complexities of systemic enterprise steering and governance.

The problem of keeping a set of accurate representations (*i.e.* models) of an enterprise is not an easy one for large enterprises, which can have hundreds of transformation initiatives, of different scopes and sizes, each year. The origin of this difficulty is that the planning process originates in many of the enterprise's communities without a consistent, coherent and complete systemic view of the enterprise to support them, individually and as a whole.

Based on our observations, the design of the enterprise's architecture is a distributed process in most enterprise, performed by architects from different domains (from business to technology), each planning, designing and deciding based on partial and often conflicting or even incompatible views of the enterprise. This state of affairs is entirely different from the one it is observed in some industries, in particular, those for which real-time systemic integrity is essential to their survival, such as Defense, where all the efforts are made to ensure at all times the enterprise transformations are steered by a centralised and fully controlled design processes[32].

The list of reports on failed *EA* programs in enterprises is large [19, 26, 14]. Nevertheless, in spite of high levels of *EA* programs failures, enterprise transformations continue to lack purposeful designs, according to systemically coherent architectures! In fact, most enterprises today are still the result of the "natural evolutions" that happened from their original birth configuration and present transformations keep being architected by enterprise architects according to partial needs and points of view, regardless of the eventual existence of global *EA* programs to guarantee the global integrity of the *TO-BE* enterprise that eventually emerges from such efforts.

What *EA* programs fail to produce is not the architecture of an enterprise, but the establishment of the processes and tools to provide complete and accurate information that sustain a good architecture design, where alignment between the different elements was taken into consideration and resulted from a conscious design. In other words, they can not create a single, consolidated view of the enterprise because this requires coordination between different areas, from coherent and non-conflicting goals, rational and implementable use of resources, cost sharing, work methods, languages, tools, *etc* [19, 26].

Such a consolidated model can be obtained and sustained by forcing all architects to use and share the same modeling notation, the same level of detail, and probably also the same modeling tool. However, this is very hard to achieve in real enterprises, because enterprise architects are in fact a very heterogeneous group, in what concerns their backgrounds, their knowledge, and needs. In practice, different architects can use different notations, different levels of detail or

even different tools, making harder to support a design process that enables consistent and coherent views of the enterprise architecture.

With the introduction of the concept of EC, we can decouple the problem of design from the problem of building a consolidated set of representations of the enterprise. Different architects can use their preferred design approach, leaving up to the cartographer the task of consolidating each partial architecture into a single global set of maps representing the enterprise architecture of the *AS-IS* as well as of the *emerging AS-IS*. In the course of this consolidation process, many inconsistencies may arise. However, they should pass on to the Enterprise Architects involved so that any inconsistencies and opposing points of view can be dealt with the adequate governance mechanisms, which form the essential core of the entire organizational governance[15]. As a matter of fact, enterprises do not have explicit governance mechanisms aimed at maintaining global systemic integrity, while subject to the multitude of changes that are in progress in a distributed fashion throughout the enterprise. To realize such governance mechanisms, they need a systemic, coherent, and complete view of the enterprise as provided by *EC*.

Another important aspect is that *EC* is less intrusive than *EA*, and therefore, more easy to implement in real enterprises, the reason being that defining the shape of the enterprise is naturally perceived as a more relevant role than representing it. Enterprise architects are naturally more willing to cooperate with others when the ownership of the design remains theirs than with other architects with whom they would have to negotiate and share the ownership of new designs.

Enterprise *AS-IS* and *emerging AS-IS* are used for different purposes. The first is a key asset to operate the enterprise, for example, to analyze the impact of a response to an event, whereas the second is a key asset for the planning of the transformation initiatives. For example, consider that today's date is January and one want to plan a transformation initiative that is expected to start in April and to conclude in September. Assuming there are other ongoing transformation initiatives, the enterprise might change from January to April, and therefore the *AS-IS* state in January is not sufficient for the planning of the actions that will occur in April. In fact, to do the job properly, one needs to keep estimating as best as possible the enterprise future *TO-BE* states from April to September, based on the best available *AS-IS* at any point in time during the transformation time frame.

Transformation initiatives create, remove or change artifacts and their interdependencies that might result in changes to Architecture, as stated in the description of their outcomes. In IT, where transformations initiatives are typically called projects, such descriptions are either stated in natural language or models such as BPMN[13], ArchiMate[25], UML[6] amongst others. The conciliation of all these descriptions in an integrated and consolidated vision is a task that entails a massive human effort and is therefore far beyond the reach of the vast majority of enterprises. Thus, the key motivation for *EC* is to provide architectural maps of the enterprise *current AS-IS* and *emerging AS-IS* with the

necessary architectural information to those planning and executing the changes. We claim that *EC* allow us to provide a generic and systemic approach, where only a minimum effort is necessary to create and maintain *current AS-IS* and *emerging AS-IS* architectural maps.

## 3 Approaches to Enterprise Cartography

The idea of creating maps from observation of the enterprise reality is not new, in particular if one considers data extracted from systems in an efficient manner to observe the enterprise reality.

At the infrastructure level, Configuration Management Databases (CMDB) as defined by ITIL [33] represent the configurations and relationships of the IT infrastructure components. Such information can be used to produce architectural maps of the enterprise infrastructure. For example, some solutions provide auto-discovery techniques that detect nodes, virtual machines, and network devices to create infrastructure architectural views. Auto-discovery is a cartographic process and requires that the type of the concepts to be discovered be known in advance[9]. The resulting CMDB instance is a partial model of the enterprise's infrastructure. This model can be communicated through different visualization mechanisms, such as textual reports or graphical views that require a symbolic notation and design rules [20].

At the business and organizational layers, cartography techniques are found within the discipline of business process management, especially in process mining. These techniques make use of event logs to discover process activities, control and data flow, and also to assess the conformance of existing processes against constraints [5, 1, 2]. Mined processes correspond to the actual instances of processes, not to the designed or modeled processes. Another example of enterprise cartography is the inference of inter-organisational processes based on EDI event logs [8] .

Business intelligence techniques that collect data from enterprise systems to produce reports and dashboards may be considered yet as another example of cartography, even though it is mostly concerned with the analytic aspects rather than with architectural (relational) aspects of the enterprise. However, we can still envisage many analytical views within an architecture, and therefore we may also consider business intelligence techniques as another example of cartography, and the traditional ETL (Extract Transformation and Loading) process as a cartographic one.

One may say that *EC* has been a reality in many specific domains, concerning specific variables and hard-coded with limited and pre-defined meta-models and concepts. Most commonly, referred Techniques aim at producing *current AS-IS* views of the enterprise. Even though not so common, some also address the *emerging AS-IS* views of the enterprise, especially the analytical approaches.

We aim at a generic and systemic approach, where the effort to produce and maintain such *current AS-IS* and *emerging AS-IS* enterprise architectural views is kept to a minimum.

## 4 Enterprise Cartography Principles

We start by presenting key definitions upon which we then present our cartography principles.

### 4.1 Definitions

**Enterprise meta-model** refers to a set of artefact types and set of allowed relations between them used in the definition and visualisation of the organisation architecture. It establishes the subjects (artefacts), the verbs (relations) and the states (adjectives) used to express the architecture. Examples of artefact types are: *department*; *information system*; *service*; *process*. Examples of relations are: *realizes*, *uses*, *etc*. Examples of states are: *productive* or *decommissioned*. We are only considering artifact types, relations and states that are relevant to describe the enterprise's architecture.

***Architectural Statement.*** An architectural statement is a well-formed proposition regarding the organisation architecture. It must necessarily be expressed using the artefact types and relations defined in the meta-model. Using the above examples, the following are architectural statements: "Sales *department uses* CRM *information system*" or "*process* complain management realizes *service* customer complain". Architectural statements can also refer to artefact's states as for example: "CRM *information system* is decommissioned".

***Architectural map.*** Since we are considering only the artifacts that are relevant for the enterprise architecture, architectural maps are simply a graphical representation of a set of architectural statements over a period of time. Having multiple statements referencing different points in time implies that that architectural maps should allow for the visualization of changes in artifacts, their relations, and states.

***Alive Artefact.*** An alive artifact is a productive one, in the sense that it can play roles in the organizational transactions and processes to create value. This means the artefact is reacheable via the dependecy graph of enterprise alive products and services. So, the aliveness state of an artifact is defined from a graph traversal and is not an intrinsic property of the artifact. The roots of this graph are usually the enterprise products and services that are active at a given moment in time.

***Transformation initiative*** is a set of planned and purposeful activities that might yield changes in the enterprise artifacts, such as projects. A transformation initiative holds a set of architectural statements to become true after its successful completion. We consider ongoing transformation initiatives to be alive artifacts of the enterprise. It is not within the scope of this chapter to fully justify this assumption. Suffice is to say that ongoing initiatives are an observable part of the enterprise reality. Depending on the enterprise meta model, the following might be transformation initiatives: the creation of a new department; the deployment of a new information system; the hiring of a new employee.

***Enterprise observation.*** This concept relates to the ability to formulate architectural statements from the observation of the enterprise reality, including both the aive artifacts as well as the descriptions of the transformation initiatives.

***Knowledge Base,*** hereafter designated as *KB*, is the repository holding the meta-model, the architectural statements and the definition of the conceptual maps

***AS-WAS* state** of the enterprise is the set of all alive artifacts, their relations and states as observed at a particular point in time in the past.

***AS-IS* state** of the enterprise is the current set of all alive artifacts, their relations and states.

***emerging AS-IS* state** of the enterprise is the set of all alive artifacts, their relations and states as observed after the sucessfull completion of ongoing transformation initiatives. If we consider that the date of completion of the last transformation in the transformation pipeline is lastDay, then the emerging *AS-IS(Td)* corresponds to *TO-BE (Td)* from any day *Td* between the present day to lastDay. That is, the emerging AS-IS corresponds to TO-BE assuming that there are no new decisions and that the ongoing changes proceed as planned.
taken into account that ongoing transformation initiatives fulfill their current plans. If the The *emerging AS-IS* state is the same as the *TO-BE* at the time of last

***TO-BE* state** is the set of alive artifacts, their relations and state in point in time in the future. The *TO-BE* considering only the outcomes of transformation initiatives corresponds to the emerging *AS-IS*.

***Enterprise Architect.*** A person who makes architectural statements regarding some point in time in the future.

***Enterprise Cartographer.*** A person that collects architectural statements from observations of the enterprise reality and produces architectural maps.

### 4.2 Principles

Our approach for *EC* is grounded on the following principles:

**Principle 1 Transformation initiatives are enterprise artefacts.**
A transformation initiative is a key artifact type in the meta-model. Thus, its architectural statements can be observed in the enterprise reality assessment and pinned down to a given point in time. Making the transformation initiatives an enterprise artifact is in line with most architecture guidelines such as ISO 42010 [18], or with the Work Package concept in both TOGAF[39] and ArchiMate[25].

**Principle 2 Changes in the set of alive artifacts are planned ones**.
This means that an artifact does not become alive or non-alive randomly, but only as a result of purposeful transformation initiatives, whose plans includes architecture statements that lead to that state. Therefore, one can foresee the set of alive artifacts at some point in time in the future by consolidating

the AS-IS with the architectural statements in the transformation initiative whose completion date precedes the desired moment in time.

**Principle 3 All enterprise artifacts have a five-state life-cycle: conceived, gestating, alive, dead, retired.**

These states and their fundamental transitions are presented in figure 1.

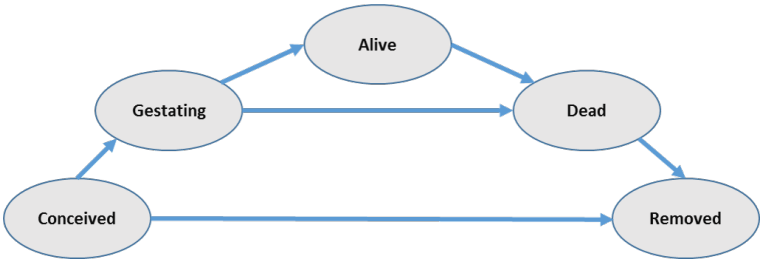


**Fig. 1.** Fundamental Artefact Lifecycle

***Conceived*** state, as the first state of existence, It corresponds to a state where the artifact is planned but before it's gestation starts. A conceived artifact can also be inferred from statements in the transformation initiatives that have not started yet. For example, if a planned initiative holds a statement to put an artifact into production at some point in the future, that artifact is in the conceived state today.

***Gestating*** is the state when an artifact is being constructed or acquired to become alive. It only differentiates from conceived state by the fact that the transformation initiative that aims at putting the artifact into production has already started. Much like conceived artifacts, gestating ones do not yet exist as an alive element of the enterprise but can be referred by alive artifacts.

***Alive*** . As defined before, an artifact is alive when it can play roles in the organizational transactions and processes to create value. From Principle 2, an artifact cannot be brought into existence as live. It always exists first as conceived of gestating before being alive.

***Dead*** is when an alive artifact is no longer able to play roles in the organizational transactions and processes to create value. As in the conceived and gestation states, a dead artifact may still be referred by alive artifacts. In fact, even if it does not create behavior or value to the enterprise, it may be the target of several housekeeping activities that are necessary after being dead.

The dead state can be achieved directly after gestating state, without becoming alive. An artifact planned to become alive by a given transformation initiative might never be alive either because the initiative was canceled or because it simply changed plans and decided to no longer put that artifact into production.

***Removed*** represents the post-dead state where the artifact has no impact in the remaining artifacts. A retired artifact is unable to interact with alive enterprise artifacts. An artifact can move from conception directly to removal when it never materialized in a gestation, meaning that it never went beyond an idea.

**Principle 4 The *TO-BE* state precedes the *AS-IS* state.**
Since artifacts in the *AS-IS* enterprise state are alive ones, they exist first as conceived ones in the *TO-BE* state, then as gestating ones in the *emerging AS-IS* state, and only after they can exist as alive artifacts in the *AS-IS* state of an enterprise.

**Principle 5 The emerging AS-IS can be inferred by observing the AS-IS of an enterprise**.
The emerging AS-IS state differs from the AS-IS state by the artifacts planned to be brought into production by ongoing transformation initiatives. Since ongoing initiatives are alive artifacts, the architectural statements that exist in their plans are in the scope of the cartographer observations of the enterprise reality.

## 5 Enterprise Cartography in Practice

In this section, we present the general steps for enterprise cartography projects that we have been using for many years now, even though initially we did not refer to them as such [46, 31, 43].

The purpose of an *EC* project is to deploy processes and tools to produce up-to-date architectural maps of the enterprise architecture with near zero effort. In the following figure, we can see an illustration of the intended projects, using the *EAMS/IT Atlas* [41] tool as the *KB* and Architectural maps generator.

The upper part of the figure 2 illustrates a scheduling of projects, where one can see that project $X$ execution is scheduled to begin and end at $Tm$ and $Tn$ respectively, and project $Y$ is expected to end between those dates. The lower part of the figure shows the *EA KB* receiving information from various sources of information and producing architectural maps. Each map has a time slider that lets you see how its contents evolve over time. By pre-assigning a color to each artifact lifecycle stage, one may see the lifecycle state of the artifacts appearing in the map at any point in time.

Consider now that project $X$ intends to add a new artifact, and that project Y aims at replacing one artifact by another. Since project $Y$ ends before project $X$, the architects of project $X$ should take into account the replacement done by project $Y$.

By loading project $X$ plans, architects can see the impact of this project in the enterprise architecture in the generated maps. The left corner of the figure 2 shows a single map with the time slider at $Tm$ and $Tn$ positions, corresponding to project $X$ begin and end dates respectively. The map contents show the impact of all the projects that finish its execution until the date selected in the time slider. So, when the time slider is set to $Tm$, the map shows the component

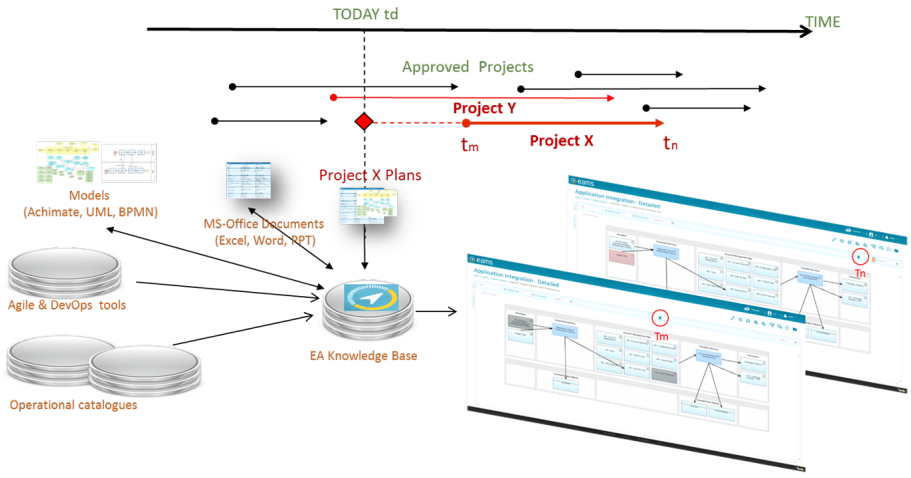


**Fig. 2.** An overview of the intended *EC* project

to be created by project $X$ and $Y$ as under-development (grey) and, when the time slider is set to $Tn$, the map shows created artefacts as alive (light blue) and project $Y$ removed artefact as dead (red).

The time slider can also operate in gap analysis mode, in which it presents the evolution of each artifact between the dates selected in the time slider. In the following figure 3, we can see the same map in gap mode between the time $Tm$ and $Tn$. The left part of the artifacts is colored with the state of the artifact in $Tm$ and the right part of the state in $Tn$. This analysis can also be done in depth taking into account the dependency graph of each artifact.

But it is clear that KB can not be fed only from the promises (plans) of the projects. At least at the end of each project, they should upload their revised plans to the artifacts that were actually produced.

As an example, consider again project X, which plans include putting into production an artifact on the date $Tn$. This promise is made at the beginning of the project by setting the alive date of the artifact to be created to $Tn$. The loading of this artifact at the beginning of the project (before $Tn$) is always a promise regarding the artifact aliveness because it refers to a future event. But, loading this same artifact on the date $Tn$ is no longer a promise but a statement about the reality, for it refers to the present.

Thus, for objects whose plans were fulfilled, nothing needs to be done. In fact, loading them again at the end of the project is an operation that does not affect the state of the $KB$, since one is only repeating an architectural statement that was already in the $KB$.

For the remaining artifacts, the revision of the project plan leads four possible situations:

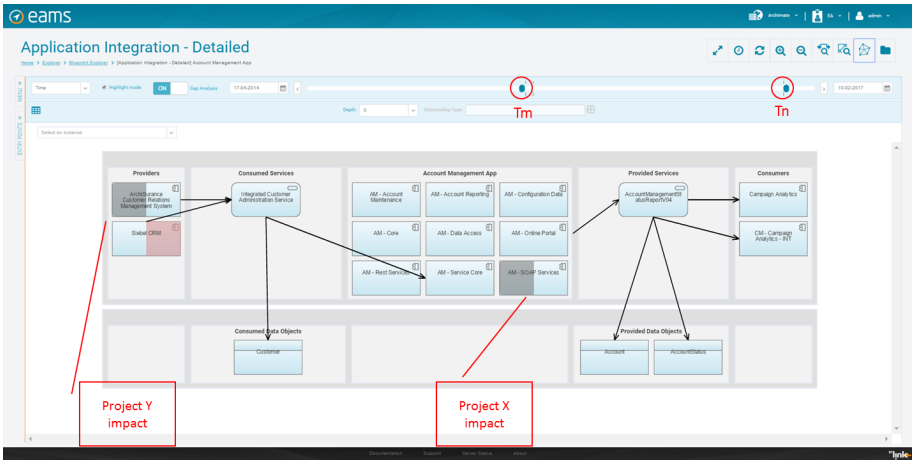


**Fig. 3.** The time Slider in Gap mode in an Architectural map example

- For the artifacts that were made productive without known and explicit plans, one needs to add the artifact with the alive date set to *Tn*, being clear that the object has a date of aliveness but does not have a date of conception or gestation. This does not mean the artifact was not conceived before, it just means their conception information and date is unknown.
- For the artifacts that were to be discontinued in the plan but they remain in production, one needs to clear their date of death.
- For the artifacts that fail to born at *Tn* as initially planned, one needs to set the date of death to *Tn*. This artifact has a date gestating and a date of death but no date of aliveness. Corresponds to what in nature is known as abortion.
- Finally, for the artifacts that were to be decommissioned in *Tn*, but remained in production, one just needs erasing the *Tn* value from the date of death.

In the rest of this chapter, we will focus on the issue of information capture, which has many issues that need to be sorted out, to ensure the project success. We start by presenting a vision of the activities of a typical *EC* project, followed by the main risk factors associated with them.

### 5.1 A template for Enterprise Cartography Projects

*EC* projects are consultancy projects developed together with the clients *EA* unit. Nowadays, these projects involve an effort between 30 and 40 men * day and have an average duration of 2 to 3 months, with the agenda of the client's senior elements having to be involved the primary factor that determines this period. These numbers do not include the effort in Phase - 6 regarding data cleansing.

An *EC* project is structured around the following phases:

**Phase 1 - Identify project goals** In general, the number of types of artifacts that can be qualified as relevant to include in the meta model is quite large. So, it is fundamental to bound the problems that we intend to respond to with the *EC* project, as a way to limit the complexity of the meta-model to the minimum strictly necessary. So, this stage boils down to a set questions whose answers the *EC* project should provide.

**Phase 2 - Define the meta-model** After limiting the scope of the project, we can define the meta-model by identifying the most relevant types of artifacts and their interdependencies required to produce the desired answers. This is an entirely straightforward exercise that starts with a standard meta-model (TOGAF Content Model, ArchiMate, among others) and extends or reduce it as required. The result is the definition of the knowledge base meta-model, and consequently, the set of possible architectural sentences one needs to capture into the knowledge base to be able to provide the expected answers.

As enterprises evolve, the KB meta-model also evolves and follows the maturity of the enterprise in these domains. Traditionally, evolution is in the sense of extending the meta-model to new areas, with more detail and with more sources of information [27].

Thus, the meta-model must be thought in such a way that it can easily be extended by areas, as the TOGAF Content Model allows it, foreseeing enrichment in different areas [39].

**Phase 3 - Identify the best sources of information** Once the artifacts types and their possible interdependencies are known, one needs to identify the sources of information that allow the collecting of information about the artifacts as effortless as possible. For each artifact, we identify the best information sources to capture state changes in the artifact's life cycle. Depending on the artifacts, one might consider a broad set of information sources.

Structured repositories systems are natural candidates for alive artifacts. For example: products price list, staff in the payroll system, application services in the services bus platform, applications in the helpdesk support system, and so on. Such systems can be a trustful source of the AS-IS, but they rarely hold information regarding the emerging AS-IS. An exception are the systems supporting the IT developing and deploying process, namely those supporting Agile development paradigm and DevOps tools, such as Jenkins[50] or Sonar[49]. Another exception is the test and quality environments since artifacts being tested will likely become alive.

Plans for transformation initiatives are possible sources of the *TO-BE* architecture. Such plans are often found either in office documents or in models in some structured notation/language, such as UML, ArchiMate, BPMN or others. In the first case, Office documents hold mostly natural language statements or images that cannot be automatically processed. For example, "the log of a board meeting where it was decided to start selling the product X in 6 month time", or "to replace a system X by system Y in the next eight months".

In the second case, the meta-models of such notations used are likely to be different from the one used in the *KB*, both in the artifact types as well as the relationship types. Thus, it is necessary to define the mapping rules to use during importation and exportation. A real example was the case were the *KB* has the *Platform* and *System Software* artefacts types. However, the models were produced in ArchiMate, where artifacts of both *Platform* and *System Software* types were modeled as instances of *System Software*, being distinguished by the fact that *Platforms* artifacts always aggregate more than one *System Software* artifact, and the latter does not aggregate other System Software. Thus, one had to load them as *Platforms* or as *System Software* depending on the defined rule outcome.

**Phase 4 - Structure the Processes and Tools to Capture Information** *EC* projects are usually conducted by the enterprise's architecture unit with the participation of other units or roles, such as project managers, technical leaders, project architects or solution architects, acting both as a consumer (stakeholders) as well as a producer. In fact, they consume architectural maps and produce the information necessary to generate them.

In our experience, the participating teams are quite skeptical about the possibility of generating updated maps without a significant effort to capture the necessary information. Thus, one has to ensure that the *EC* project does not require the remaining actors to do more work than they already have today. Our approach is, firstly, to look at the tools the teams use to produce and register architecture relevant information and propose the necessary changes so that it can be automatically loaded into the Knowledge base. This approach allows the loading of information and the production of maps for the teams to use with a minimum negative impact on them. For example, in an enterprise that uses MS-Office documents for internal project presentation and description, we can include tables in those documents to automatically load the information (the architectural statements) about such project. The figure 4 shows an example for project *X* (used as an example earlier), with a first table for general data and a second for applications components affected, accordingly to the map shown in the previous figure 3.

Later, after the remaining teams are using the maps generated with the information collected, we can propose more structured ways of obtaining the required information, either through the models or direct introduction in the *KB*.

**Phase 5 - Define and Configure the Architectural Maps** The definition and configuration of the intended maps are fundamental for their adaptation to the needs of the various stakeholders. The maps generated are hierarchically structured since, in our experience, they tend to be one of the preferred ways to visualize architectural maps. This is also confirmed in [34], where such maps are named clustered maps. Despite the hierarchical structure, the map generator can be configured to produce all ArchiMate viewpoints [25]. As an example, besides

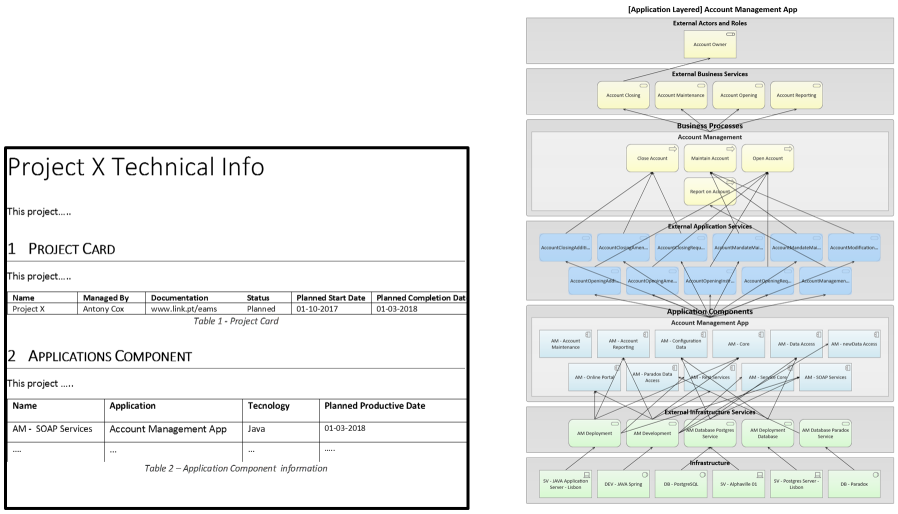


**Fig. 4.** On the left: and example of the word file to be imported. On the right: an example of a Layered Architectural Map

the integration map presented in figure 3, we show a layered map in the figure 4, corresponding to the ArchiMate Layered Viewpoint [25].

It is important to clarify that not everyone wants to deal with the complexity of the *KB* meta-model. Therefore, our architectural maps act as a *View* in a relational DBMS, allowing navigation and exploration under a perceived meta-model specifically designed to each stakeholder in that map. A typical situation is to reduce the number of entities one need to cross to find the dependency between two concepts, showing only the derived relations instead of the existing ones in the meta-model. For example, in a *KB* with the ArchiMate meta-model, users of a given architectural map could experience and navigate over simplified meta-model where business processes are directly related with applications without going through the services and functions in between.

**Phase 6 - Populate the *KB* with an initial baseline** The initial loading of the *KB* is usually done by importing other systems or even existing excel sheets that the client may have with relevant information. However, loading AS-IS can be a very challenging step in enterprises with a low maturity level on these subjects. As a matter of fact, if enterprises had their AS-IS ready to be loaded, they probably would not need an *EC* project to start with. The main problem is usually in the quality of data of the information to be loaded, as well as the lack of homogenization at the conceptual level. Concepts such as Information System, Application, Service and Platform are among the most complex to grasp.

The effort of loading information into KB is tiny. On the contrary, the work needed to prepare the information to be loaded (data cleansing) can be very high.

In practice, there are only a few catalogs that are to be loaded, in a Pareto-law approach.

Obviously, the more complete and rigorous the *KB* is, the more useful and savings one gets from the *EC* solution. Therefore, this initial loading is often perceived as an essential step to the success of the project. In our view, this initial upload is actually optional and should not be placed as a critical success factor for the project. We argue that the true critical success factor is the ability of uploading and updating the *KB* on a project by project basis. The more projects, the more rapidly the *KB* will be populated and kept updated. So, in the approach presented we will consider that the *KB* is loaded incrementally, project by project. This task can be done in parallel with the Architectural Maps configuration.

## 6 Enterprise Cartography Cases

### 6.1 Case 1 - Enterprise Cartography as a monitor to the progress of the strategic roadmap

The Serasa Experian (hereafter designated by Serasa [46]) is amongst the biggest credit bureau in the world outside the USA, holding the largest database of consumers, companies and economics groups in America Latina. The company is involved in most credit decisions taken in Brazil, corresponding to 4 million queries each day, done by over 400 thousand clients. With a strong transformation plan, Serasa needed tools to monitor the progress of the strategic roadmap.

Serasa had a strategic plan for the entire IT that is embodied in roadmaps for the key business and technologic areas of the company. These roadmaps set the targets to achieve for each area over the next five years. These targets are defined regarding a maturity level measured on a 5 level scale[21]. At the end of each year, each area submits a plan with the initiatives proposed for the coming year. If approved, these initiatives become real projects. Then, one year after, each area evaluates the progress achieved and updates the roadmap accordingly. Roadmaps were manually maintained in many Microsoft Excel sheets, requiring a significant effort to update the roadmaps according to the results of the projects.

Our task was to design, plan and implement an instrument that enables the integrated management of architectural views, project results and the roadmap progress. More specifically, this tool should allow Serasa to:

- Have an integrated and up-to-date view of the *AS-IS* of business and IT architecture, by the consolidation of information from different sources.
- Have a view of the emerging *AS-IS* state of the business and IT architecture based on the foreseen results of on-going and planned projects.
- Have the ability to identify the progress of each roadmap updated according to the projects' progression.

Following our project template, the different areas had identified 82 relevant questions, leading to a meta-model with 32 types. Each of the areas provided the

information to be loaded in the KB, both regarding the roadmaps and the reality of the enterprise at different levels: processes, skills, software and infrastructure. The architectural maps were defined in sessions with each of the areas.

The immediate result of this project is to provide to the various areas of IT a consolidated and updated view of the architecture of Serasa. Consolidated, because it results from the information provided from different areas. Updated, because it results from information sources maintained by the various areas with transparent processes in an automated manner. Since each group shared its information with other areas, the project entails a substantial transformation in the enterprise, as it homogenized languages and tools. In this regard, the *KB* meta-model is a valuable asset because it identifies the concepts familiar to the various areas. A more detailed description of this case was published in co-authoring with the architecture team of Serasa in [30].

### 6.2 Case 2 - Supporting a Prescriptive approach for Enterprise Architecture to reduce IT spending

This case presents the set up of an IT architectural *KB* for the public bodies of the Portuguese central government. The Agency for the Public Service Reform (AMA [44]), is a public institute that has the mission to roll out modernization initiatives in the administrative and regulatory areas focused in the simplification and optimization of public services, within the policies defined by the Government.

AMA had the task of achieving an *EA* at the Portuguese Central Government, aiming at a more rational IT spending within the public organisms. The project aims at the creation of a central *KB* to allow a descriptive and prescriptive approach for *EA* [12, 15] over the entire set of enterprises belonging to the Portuguese Government.

The adoption of a simple *KB* meta-model and intuitive maps to seems critical to achieving an active *EA* over such a broad scope. Even so, the meta-model contains 24 artifact types, from business to infrastructure domains. The *KB* was initially loaded with the information from a previous assessment of about 200 public agencies from 7 different Ministries. About 87K devices, 15K servers, 14K databases, plus 20 other artifacts types were loaded at this stage.

After the initial loading, the *KB* is updated on a project by project basis, since public bodies are required to submit to AMA the architecture of the IT projects over 10K €. Each submission is validated against a set of regulations applied to public bodies. Such ruling aims at the implementation of a prescriptive and normative approach for *EA* by promoting the adopting of a Reference Architecture, Design Principles and the sharing of available resources within the public administration. This project was presented in co-authoring with the AMA architecture team in [43].

The project is still ongoing in some ministries that followed a federative view of the *KB*, where the ministry has its KB with a meta-model enriched to address specific concerns in that domain, namely health and defense. First, the public bodies feed the *KB* of their Ministry, by submitting the proposed architectural

scenario and then, upon approval against Ministry own rules, it can be sent to AMA for its approval and *KB* update.

### 6.3 Case 3 - *EC* as the provider of organisation views for ITIL processes

The IT Department of the Portuguese Defense [45] (or CDD for short), provides IT services and support to Portuguese Defence forces. CDD adopted ITIL [33] processes to improve the quality of its support services. Although the processes were defined, there was still much to improve in the description of information and concepts among the different teams involved in the execution of these operations. Each technical area managed information relating to their respective functions and technical competencies. This information was not integrated or even shared, as each technical area had its own database with related information. One example was the lack of a coherent service catalog, since each technical service area had identified their services at different levels of granularity, and fail to integrate them into a single one.

The purpose of the project was to provide the CDD with a *KB* that was common to the various technical teams, homogenizing terms, concepts and ensuring the coherence of information used in the execution of ITIL processes. To structure the information sources CDD had to clarify the semantics of each concept and their relationships. They identified the information sources from each technical area in the meta-model. From the identified information, CDD developed the architectural layers: infrastructure architecture involving the servers and networking views; data architecture with databases and instances; and applications architecture.

The meta-model [11] was derived from the service orientation, reaching a model with 16 concepts. Hence, they identified the information sources from each technical area in the defined meta-model. Together with each of the four technical areas, CDD identified, clarified, and selected 12 primary sources of diverse information that need to be integrated and loaded into the repository. ArchiMate models with the ITIL processes and resources were also loaded into the *KB*. Nowadays, the information serves the CDD interest as a whole instead of relating to a unique technical area. The achieved results allowed the CDD to provide consolidated and updated information to the various technical areas through views of the architecture.

The project was fully planed and executed by CDD architecture team, including the design of the *KB* meta-model, the configuration of the loading rules from the CMDB and ArchiMate models, and the design and configuration of the architectural maps made available to the technical teams[10].

### 6.4 Case 4 - Architectural maps in Enterprise Intranet and Wikis to support development teams

The client referred in this case study is a retail bank that operates in the Iberian Peninsula and serves almost 2 million customers (Individuals, Companies and

Institutions) through its multi-channel distribution network comprising around 650 retail branches.

The bank already had an *EA* portal on their intranet to support the registration of information regarding Application and Technology Architectures. The intranet portal included a large number of features such as a wiki, with a considerable amount of articles that described the processes, system architecture, and which established an entry point for all documentation deemed more technical.

With this project, the bank was addressing several challenges. The first one was to develop the bank´s *EA* practice, by improving the *EA* intranet portal to support the communication and awareness of the architecture between all stakeholders, from infrastructure teams to business areas.

The second challenge was the need to effortlessly maintain the architectural representations. The bank had a experience in EA, since it went through the process of using a modeling tool to produce representations and models of its IT Architecture. The approach followed was based on a central repository holding all models, each designed manually with an *EA* tool. This approach required a substantial effort to keep them up-to-date, in particular if one considers the consolidation of the models produced in different projects into a single and enterprise-wide view of the IT landscape. So, the maintenance effort to keep the *EA* representations was a key concern of this project and the bank stated that "if an architectural view cannot be generated automatically with up-to-date information, then it cannot be presented in the *EA* portal". The third challenge was the need to keep audience's learning curve regarding the usage of the architecture intranet portal as lean as possible. The fourth challenge was the integration of the SOA initiative in the Enterprise Architecture Initiative.

The project aimed at the implementation and deployment of an *EA* solution fully integrated into the existing intranet, so that all but back-office housekeeping activities could be done in a seamless manner on the bank's intranet. The bank also wanted to address other architectures in domains they considered relevant, even though they did not have much information to start with, namely: Business, Information, User Experience and Normative Architectures. Both Information and Business Architectures were domains that the bank planned to address in this project, starting with the definition in the meta-model, where it became clear the level of detail needed and how to establish the relations with the remaining architectures.

The User Experience Architecture concerns with the definition of patterns to allow the same concept (client, address, account and so on) to appear to the user with the same paradigms, regardless of the interface and application where such experience occurs. Finally, the Normative Architecture structures all information regarding Architectural Principles, Rules, Best Practices and Technical Articles amongst other information. Articles represent a real knowledge base, and a wiki platform allows live interaction with end users.

The architecture views were generated on-the-fly and embedded in a seamless manner with the bank's intranet portal. Architectural views are the entry point to access any IT documentation, and knowledge kept in wikis. The information

necessary to generate the architectural maps had to be harvested from various sources, such as Microsoft SharePoint[52], Oracle Enterprise Repository[48], among others.

This approach enables not only the role of the architecture views in understanding the complexities of the business and the IT underneath, but also a better and more efficient collaboration between the different stakeholders. This has a positive feedback on the use of the *EA* internet portal as cooperation and communication platform, and it plays a fundamental role in people communication and collaboration. This case is published in[31].

### 6.5 Case 5 - Clarification of the concept of IT Application.

The case presented here focuses on the issue of standardization and structuring of project submission documents, addressed in "Structure the Processes and Tools to Capture Information" presented in the project template section.

The IT industry has multiple terms with more or less obvious meaning; words such as application, information system, and business solution tend to be used indistinctively in various scenarios and can become the source of confusion in many situations. The existence of an application/system/solution catalog is common in many enterprises, and it is considered a fundamental element that draws attention from the Business, the IS and the IT area.

With over 20.000 employees spread throughout the world and with more than 4 million clients, the Caixa Geral de Depósitos [42] (CGD for short) is the largest Portuguese financial group, encompassing banks, insurance and medical companies. The Sogrupo Sistemas de Informação (or SSI) is the company responsible for supporting all Information Technologies in the group. SSI has approximately 900 workers. The SSI has an ongoing *EA* program which enforces both the descriptive and prescriptive perspective of architecture. It follows TOGAF in its main ideas.

The project in question had the mission of clarifying the Application concept to the primary stakeholders across the SSI, and to restructure the key Application Technical Architecture in accordingly. With over to 500 so called applications to classify, the goal was not to provide formal and theory-driven definitions but instead to establish a concept that was useful and very pragmatic to be used by the multiple stakeholders involved.

As the result of the project, one realizes that the term application embodies aspects of the three different perspectives (business, information systems and infrastructure) and that it would be better to decompose the application into three distinct concepts, each covering only one perspective and with a single responsible.

- At the level of the business, the concept of application relates to functional issues. Here, the term Business Solution was the one that garnered more acceptance. The person in charge of a Business Solution decides only the functional aspects and who can access.

– At the level of information systems, the application concept unfolds in a set of components to support the defined requirements. The person in charge of an Application focuses on the architecture and engineering of these components.
– Finally, at the level of the infrastructure, the concept of application materializes in the capacity of execution of the defined components. Here, the term chosen was Platform, and the person responsible for it is fundamentally concerned with ensuring the continuity of its operation.

As a result, the architecture and catalog of an application may lead to 3 distinct artifacts, each with its domain and concern. But, of course, there is a many-to-many relationship between the artifacts of these three levels, and an application component can support different business solutions and a business solution to be supported by many application components. The same for applications and platforms. This work is published in co-authorship with Architecture responsible at CGD in [29].

## 7 Discussion

Although there are many complex issues regarding the generation of architectural maps automatically and on-the-fly, we keep the discussion focused on issues centered on information capture. We select the following topics:

– Start using the *EC* solution with the empty *KB*.
– Projects as information sources.
– Corrective and evolutive maintenance
– Business Cartography

**Start using the *EC* solution with the empty *KB*** In cases where *KB* is not loaded from the beginning, it will not be able to feed the transformation initiatives with information necessary for their planning and impact analysis. In this situation, the projects feed *KB* instead, both with the *AS-IS* as well as the *TO-BE*.

Whenever a project proposes to do some form of integration between the new artifacts and those existing in the enterprise, the analysis and estimation of the time, cost and risk associated with that integration requires some assessment of *AS-IS*. If this information does not exist in the *KB*, the effort to survey the *AS-IS* should be included in the project planning.

Therefore, at first, while the *KB* is practically empty, the projects have to consider the effort and time to do the necessary assessment of the *AS-IS*. However, as *KB* becomes loaded, projects become information consumers and take full advantages of the *KB* and related architectural maps to support their analyses and planning. The faster the enterprise changes, the more project will be loading the *KB*. After a few iterations, only *AS-IS* systems that have never been involved in any project will remain unknown.

**Project as information sources** As we mentioned, in our approach the *KB* should be uploaded project by project, to ensure that its impact is present on the maps and in this way is known by all. We argue now that each project should provide information in two key moments:

- The first one, when the project architecture already contains the main elements to become productive (alive) or decommissioned (dead), and consequently they should be loaded into the *KB* so that its impact on the other projects can be known and evaluated. The information uploaded is just a promise, that will or will not come to fruition. This moment usually occurs when the project is evaluated for go / no go decision making since it is at this time that it is necessary to assess the costs, time and risks associated with its execution.
  The reader could argue that architecture is substantially independent of cost, time, and risk issues. But our experience is just the opposite: the more rigorous the evaluation of cost, time and risk the more complete and detailed the architecture is known. In fact, this finding is perfectly aligned with the most common definition of Architecture, in which it includes knowledge that is necessary for the maintenance and transformation of enterprises [40, 24, 7, 16]. Consequently, architectural knowledge is a necessary asset to estimate costs, times and risks associated with transformation initiatives. The close relationship between the *EA* and the management key variables associated with transformation is a reliable driver to force project leaders to provide relevant architectural information early in the project lifecycle.
  Furthermore, assessing the impact on cost, time and risk management is a way of determining the appropriateness of an artifact in the enterprise's architecture. Regardless of the theoretical discussion that this line of thought can take us, the more impact an artifact has on those three variables, the more likely it will be identified and known in the plans of a project.
- The second moment is at the end of the project, since at this time the artifacts and relationships created or eliminated by the project are known for a fact. The importance of updating the project plans and reloading them in KB is obviously to differentiate what are plans (promises) from what is the reality (*AS-IS*).
  As seen earlier, artifacts that were not initially foreseen should be created with the gestation date and alive coinciding with the beginning and end of the project, respectively. These dates are automatically filled in, during the loading of the project information according to the configured rules. These artifacts are distinguished from those that were originally planned because they do not have an existence before the project. By contrast, artifacts planned from the beginning have an existence (conceived) before the start of the project.
  The artifacts that were planned but not put into production at the end of the project become dead even though they have not gone through the alive state. It is a situation of abortion, in which one passes from gestation directly to death. It is very importance to record this evolution, for as long as they were

in the gestation state, these artifacts were alive in the enterprise emerging *AS-IS* and therefore may have influenced decisions that are only justified in the assumptions of the existence of these objects. For example, consider that project Y made decisions based on the assumption that project X, in progress, would create a particular artifact. If this will never be brought into production, it is important to keep such fact in architectural maps so that project Y can justify the decisions made.
A second reason to load the project information at the project completion concerns with updating the dates related to objects lifecycle automatically. Since, most probably, projects fail to end at the dates initially planned, it is important to automate the setting of the dates of the artifacts based on the dates of the projects that create or discontinue them so that they can become correctly with no effort.

**Corrective and evolutive Maintenance** The assurance that the information residing in the KB remains up-to-date is supported by the assumptions of principle 2, in which it is assumed that *AS-IS* is portrayed in some way in the *TO-BE* view. If the evolution of the enterprise is fundamentally based on projects, the *KB* is kept up to date. However, it may happen that the evolutive and corrective maintenance also changes the enterprise architecture. Since maintenance actions tend to follow a lighter approach than projects, one must assure that architecture changes are in some manner propagated to the *KB*.

The simplification of the process involves uploading the changes made at the end of the action, thus omitting the loading phase of the plans. Assuming that maintenance actions are mostly functional, thus do not cause changes to the architecture, the need to report changes to the architecture is more an exception than the norm. Even so, it is important that maintenance teams have a list of the types of artifacts that are being tracked in the context of the *EC*. A simple list of meta-model artifacts (typically with 12 to 20 entities) allows maintenance teams to mitigate this issue.

**Business Cartography** Throughout this chapter, our focus and example have been mainly on the evolution of IT, although we consider our approach to mapping to be generic and also applicable to business transformations. But there is indeed a crucial factor that distinguishes transformations at the level of business from those at IT. It is that project management a mature discipline in most enterprises, involving virtually all changes in the IT architecture. All enterprises seek to know the time and risk costs of IT projects, thus sustaining our approach. On the contrary, transformations at the business level often happen without a project being formed. A typical example is the creation of a new department, which can arise only from the course of a series of meetings without any TO-BE document or explanatory model. This is a difficulty that needs to be mitigated, for example by defining structured templates for meeting minutes so that they can be uploaded and can be valid sources of information.

## 8 Conclusions

Having discussed the most critical points of information gathering in enterprises, in this conclusion we explore the role of *EC* in the Governance and Organisational Self Awareness (OSA) [36, 3, 35, 37, 38].

The formal and rigorous emergence of the discipline of Enterprise Engineering, scientifically based on System's Theory [55] and solidly grounded on Social Sciences Theories, has provided a fertile ground for interdisciplinary research involving key aspects of the full lifecycle of dealing with Enterprises as complex artifacts populating and serving the Human habitat. The general engineering lifecycle of Enterprises involve aspects of Architecture, Engineering, Governance and Dynamic Steering of the time transformations that always occur in all living entities. The present paper focused the role and relevance of *EC* and Enterprise Cartographers and clarified the distinction between those and *EA* and Enterprise Architects.

In a previous paper entitled "The Role of Enterprise Governance and Cartography in Enterprise Engineering" [35], a more general view of *EC* was presented, by taking into consideration the view of Enterprises as organized complexities, whose existence at any point in time is the result of the dynamic synergies between purposeful, intentional, goal-oriented designs, such as those generated by *EA*, with the emergent, opportunistic, individualistic, non-orchestrated and often incoherent and chaotic actions that naturally occur as the result of the free will of the human actors that are, after all, the true core of any organisation. We conclude this paper by connecting the dots between purposeful *EA* designs leading to intended pre-determined Enterprise Transformations and the realities of real-time events, with the emergent phenomena that in fact end up being the reality of the enacted enterprise as *de facto* is.

The key to connect, integrate and make sense of these two powerful reality drivers, one more top-down, the other clearly bottom-up, is precisely the capability to sustain, at the systemic organizational level, commonly shared representations of the *AS-IS* of perceived reality as accurate as possible. We have coined the concept of "organizational Self Awareness (OSA)" to denote this systemic capability, whose fundamental technical basis is precisely the one addressed in this paper: Enterprise Cartography.

By collecting the sensory data of the enacted reality as it happens and is perceived by those that are involved and immersed in the respective action context; by feeding such data into semantic organisational models capable of being read, understood and interpreted by the organisational actors, according to their multiple and specific points of view; and by providing the dialectic contradictory mechanisms to challenge, verify and consolidate the sensory data being reported as true, and systematically building a verifiable and verified explicit knowledge base of the enterprise reality and its dynamics on-the-fly, one is providing to all involved – from the lowest level of operational workers to the top level of enterprise executives and strategists – a formal, updated, common view of reality which is key to sustain an essential feature of any organisation: meaningful, in-

formed, objective, explicit, non-ambiguous means of support to human-to-human communication.

The capability of an organisation to be aware of itself at any point in time, in space and in context, is the equivalent to what we, single humans, do possess as most valuable to steer ourselves in life: our own self-awareness, on top of which we build higher order levels of self, such as conscience, values and ethics. *EC* is, at the very root, the basic mechanism for us to equip modern day organisations with the artefacts we need to deal with the tremendous challenges we face today: exploding complexity, increasing speeds of interactions, gigantic volumes of information flows, shorter and shorter decision times and increased requirement to build up on-the-fly agile and realizable answers to previously unknown challenges.

## Acknowledgements

This work was supported by the project 11419 (IT-Atlas) of IAPMEI, under the 2020 Portuguese PO CI Operational Program. The authors would like to thank André Sampaio e Ricardo Leal from Link Consulting SA that contributed for most of the projects presented here.

## References

1. Aalst, W. Van der. (2011). "Process Mining: Discovery, Conformance and Enhancement of Business Processes", Springer-Verlag.
2. Aalst, W. Van der, *et. al*, (2012). "Process Mining Manifest". Business Process Management Workshops. Lecture Notes in Business Information Processing Volume 99, pp. 169-194.
3. Abraham, R., Tribolet, J., Winter, R. (2013). "Transformation of Multi-level Systems – Theoretical Grounding and Consequences for Enterprise Architecture Management", Advances in Enterprise Engineering VII , Third Enterprise Engineering Working Conference, EEWC 2013, May. 2013 , pp. 73-87 , Springer.
4. Adams S. (2009). "ITIL V3 foundation handbook". Vol. 1. itSMF International, Office of Government Commerce, ISBN: 978-0113311972
5. Agrawal R., Gunopulos D., Leymann F. (1998). "Mining process models from workflow logs". Springer.
6. Booch, G., Rumbaugh, J., Jacobson, I. (2005). "The Unified Modeling Language User Guide". Addison-Wesley Professional.
7. Dietz, J. (2006). "Enterprise Ontology - Theory and Methodology", Springer.
8. Engle, R., Wil M. P. van der Aalst, M. Zapletal, C. Pichler and H. Werthner. (2012). "Mining Inter-organisational Business Process Models from EDI Messages: A Case Study from the Automotive Sector". Advanced Information Systems Engineering, Lecture Notes in Computer Science Volume 7328, pp. 222-237
9. Filipe J. A. C. B., Coelho M. F. P., Ferreira, M. A. M., Figueredo I. C. (2011) "Social Responsibility and Environmental Sustainability: The Case of Caixa Geral de Depósitos (Portugal) and Vale (Brazil)". In: Editorial Board Members 10(5), pp. 352– 375

10. Gama, N., Sousa, P., Mira da Silva, M. (2015). "A case of integration between ITIL and TOGAF". In proceedings of the 8th Workshop on Transformation & Engineering of Enterprises (TEE 2015) (in CBI2015) -14 July, Lisboa, Portugal.
11. Gama, N., Mira da Silva, M., Francisco, R.A. (2011). "A Conceptual Framework for Structuring an IT Enterprise." 16th Annual Conference of the UKAIS, Oxford, England.
12. Greefhorst, Danny, and Erik Proper. (2011). "Architecture principles: the cornerstones of enterprise architecture". Springer Science & Business Media.
13. Grosskopf, Decker and Weske (2009). "The Process: Business Process Modeling using BPMN". Meghan Kiffer Press.
14. Gunther, W. (2014). "Measuring Enterprise Architecture Effectiveness". Leiden University: Leiden University (online).
15. Hoogervorst J. A. (2009) "Enterprise governance and enterprise engineering". Springer.
16. IEEE, (2000). "The IEEE Standard 1471-2000 Systems and software engineering — Architecture description". IEEE Computer Society. superseded by ISO/IEC/IEEE 42010:2011.
17. IEEE. (2010). "Systems and software engineering – Vocabulary", in ISO/IEC/IEEE 24765:2010(E) , vol., no., pp.1-418, Dec. 15
18. IEEE. ISO/IEC/IEEE 42010:2011. (2011). "Systems and software engineering - Architecture description". ISO.org. November 24.
19. Jacobson, I. (2007). "Enterprise Architecture has failed in a big way!" www.techrepublic.com/article/enterprise-architecture-has-failed-in-a-big-way (online)
20. Lankhorst, M. (2013). "Enterprise Architecture at Work: Modelling, Communication and Analysis". The Enterprise Engineering Series, Springer. ISBN: 978-3-642-29650-5
21. Maier, A. M. , Moultrie, J. and Clarkson, P. J. (2012). "Assessing organisational Capabilities: Reviewing and Guiding the Development of Maturity Grids". in IEEE Transactions on Engineering Management, vol. 59, no. 1, pp. 138-159, Feb.
22. Le Moigne J.-L. (1977). "La theorie du systeme general". Theorie de la modelisation, le Moigne J.-L. (ed.), 2006th ed. Collection Les CLASSIQUES DU RESEAU INTELLIGENCE DE LA COMPLEXITE
23. Morris, C.W., (1938). "Foundation of the Theory of Signs", International Encyclopedia of Unified Science, Vol. 1, No. 2.
24. Op't Land, M., Proper, E., Waage, M., Cloo, J., Steghuis, C. (2009). "Enterprise Architecture Creating Value by Informed Governance". Springer.
25. The Open Group. (2015). "ArchiMate® 2.1 Specification". Van Haren Publishing, Zaltbommel, www.vanharen.net.
26. Roeleven, S., Sven, and Broer, J. (2010). "Why Two Thirds of Enterprise Architecture projects Fail". www.cio.com.au/whitepaper/370709/why-two-thirds-of-enterprise-architecture-projects-fail/ (online).
27. Silva, N., Ferreira, F., Sousa, P., Mira da Silva, M. (2016). "Automating the Migration of Enterprise Architecture Models". International Journal of Information System Modeling and Design (IJISMD) 7.2 pp 72-90.
28. Sousa, P., Lima, J., Sampaio, A., Pereira, C., (2009). "An Approach for Creating and Managing Enterprise Blueprints: A case for IT Blueprints". The 21st International Conference on Advanced Information Systems, Lecture Notes in Business Information Processing, vol. 34, pp. 70–84, Springer-Verlag, The Netherlands .

29. Sousa P., Martins R., Sampaio A. (2012). "A Clarification of the Application Concept: The Caixa Geral de Depósitos Case". In: Proper E., Gaaloul K., Harmsen F., Wrycza S. (eds) Practice-Driven Research on Enterprise Transformation. PRET 2012. Lecture Notes in Business Information Processing, vol 120. Springer, Berlin, Heidelberg
30. Sousa P., Gabriel R., Tadao G., Carvalho R., Sousa P.M., Sampaio A. (2011). "Enterprise Transformation: The Serasa Experian Case." In: Harmsen F., Grahlmann K., Proper E. (eds) Practice-Driven Research on Enterprise Transformation. PRET 2011. Lecture Notes in Business Information Processing, vol 89. Springer, Berlin, Heidelberg
31. Sousa P., Sampaio, A. Leal, R. (2014). "A case for a Living Enterprise Architecture in a Private Bank", In proceedings of the 8th Workshop on Transformation & Engineering of Enterprises (TEE 2014), July, Geneva, Switzerland.
32. Steven H Dam. (2015). "DoD Architecture Framework 2.0: A Guide to Applying Systems Engineering to Develop Integrated". CreateSpace Independent Publishing Platform.
33. Office of Government Commerce. (2007). "The Introduction to the ITIL Service Lifecycle". The Stationery Office (TSO), London, UK
34. Roth, S., Zec, M., Matthes, F. (2014). "Enterprise Architecture Visualization Tool Survey" 2014. Technical Report. sebis, Technische Universitat Munchen.
35. Tribolet, J; Sousa, P.; and Caetano, A. (2014). "The Role of Enterprise Governance and Cartography in Enterprise Engineering". Enterprise Modelling and Information Systems Architectures, 9 (1): 38-49
36. Tribolet, J. (2005), "Organizações, Pessoas, Processos e Conhecimento: Da Reificação do Ser Humano como Componente do Conhecimento à Consciência de Si Organizacional", in Sistemas de Informação Organizacionais, Chapter, Nov 2005, Edições Sílabo - Ed. L. Amaral.
37. Tribolet J. (2014). "An Engineering Approach to Natural Enterprise Dynamics - From Top-down Purposeful Systemic Steering to Bottom-up Adaptive Guidance Control" / J. Tribolet - ICEIS 2014 keynote.(online) http://www.iceis.org/KeynoteSpeakers.aspx?y=2014#4
38. Tribolet, J. ,Pombinho, J., Aveiro, D. (2014) "Organizational Self Awareness: A matter of Value". Organization Design and Engineering: Co-existence, Co-operation or Integration, 146-176, Dec 2014, Palgrave Macmillan. pp 146–176,
39. Van Haren. (2011). "TOGAF Version 9.1". (10th ed.). Van Haren Publishing.
40. Zachman, John A. (1997). "Enterprise architecture: The issue of the century". Database Programming and Design 10.3. pp. 44-53.
41. https://eams.linkconsulting.com, accessed in September 2017.
42. https://www.cgd.pt, accessed in September 2017.
43. http://www.ciaonetwork.org/uploads/eewc2014/industrial_track_talks/pedro_sousa_and_andre_vasconcelos.pdf, accessed in September 2017.
44. https://www.ama.gov.pt, accessed in September 2017.
45. http://www.defesa.pt, accessed in September 2017.
46. https://www.serasaexperian.com.br, accessed in September 2017.
47. https://www.merriam-webster.com, accessed in September 2017.
48. http://www.oracle.com/technetwork/middleware/repository/overview/index.html, accessed in September 2017.
49. https://www.sonarsource.com, accessed in September 2017.
50. https://jenkins.io, accessed in September 2017.
51. https://en.wikipedia.org/wiki/Cartography, accessed in March 2017.

52. https://products.office.com/en-us/sharepoint/collaboration, accessed in September 2017.
53. https://en.wikipedia.org/wiki/Nyquist_rate, accessed in September 2017.
54. https://en.wikipedia.org/wiki/Control_theory, accessed in September 2017.
55. https://en.wikipedia.org/wiki/Systems_theory, accessed in September 2017.
56. https://en.wikipedia.org/wiki/Dynamical_systems_theory, accessed in September 2017.