Enterprise Methodology: an Approach to Multisystems

Dominique VAUQUIER

Praxeme Institute, Group IT Strategy & Enterprise Architecture team at AXA Group

Cell: +33 (0)6 77 62 31 75

dominique.vauquier@praxeme.org

www.praxeme.org

Abstract: Praxeme is an open method backed by many companies and organizations. Its scope is the "enterprise", understood in its most generic and widest meaning. The methodology is a tool – sometimes a weapon – that helps us to cope with the complexity of the objects we want to create or to transform. The Enterprise System Topology provides us with a methodological framework that identifies and links the aspects of the enterprise together. Based on this framework, the convergence theory describes the way of transforming a collection of companies into a cohesive and efficient single system. The range of topics includes business knowledge, organization and IT.

Methodology, transformation, enterprise, complexity, merge, convergence, system, modeling

The initiative for an open method involves public organizations (French Army, Direction Générale de la Modernisation de l'Etat…) as well as private companies (AXA Group, SMABTP, SAGEM…) and consulting firms. It has resulted in the Praxeme method whose scope embraces all the aspects of the enterprise, from strategy to deployment. After a brief presentation of its principles, this paper will focus on its contribution to the transformation of the enterprise. More specifically, it will address the question of how to master the complexity of the extended enterprise and how to organize the convergence of its constituents.

The limited length of this paper and lecture makes it necessary to leave a couple of issues aside. As a result, it may sound a bit of a utopian approach. Indeed, we assert the claim: if we really are to transform the enterprise – as more and more decision-makers are proclaiming – we must be able to think it anew. That is not to say that our approach is not realistic. The methodology is precisely there for making good-will and ambition effective.

The main obstacle on the road consists in the preconceived ideas that the crowd of naysayers and ideal-killers endlessly reiterate. Answering this criticism is easy – technically speaking – but it would require more space and time for delving into the details of the method. We will only highlight the core message as far as complexity is concerned. Firstly, we will expose the methodological framework which provides the theoretical basis of the method. Then, we will apply it to the challenge of convergence inside large organizations.

The Enterprise System Topology

Notion of Enterprise System

We call "Enterprise System" the enterprise that perceives itself as a system[1]. Using such a phrase expresses a strong tenet: in the face of complexity, we adopt a specific sort of rationality, made up of scientific assessment, engineering, system theory… In the workplace, this is not such a natural posture. The Enterprise System is the entire enterprise, a complex object. It must not be confused with the notion of an IT system at the scale of the enterprise (namely a group or a federation). Beyond the IT system, it includes the numerous and various constituents of the enterprise, some of which are material – like buildings, equipment, people – others are resolutely abstract – like values, goals, knowledge… A great deal of the complexity stems from the diversity of these constituents as well as from the fact that they are hugely intertwined. To give an example, a person as a worker is an obvious constituent of the enterprise; this person assumes a role in the organization depending on his/her skills and behaves in accordance with his/her personal values. For every task undertaken, there is a potential for conflict between these individual values and the asserted or real values of the group. As a result, solutions – processes, software… – may or may not work depending on the level of harmony that has been established between the value system of the group and the one of its members. All elements cited in this example are part of the Enterprise System. Remain oblivious to these elements and our action will soon be hindered. Recognizing this reality is common wisdom.

[1] Cf. the Enterprise Transformation Manifesto (http://www.enterprisetransformationmanifesto.org).

Taking it into account in our thoughts and actions is less common and, on the contrary, it requires thorough attention and constant endeavor. That is the meaning and content of the phrase “Enterprise System”.

Methodological Framework

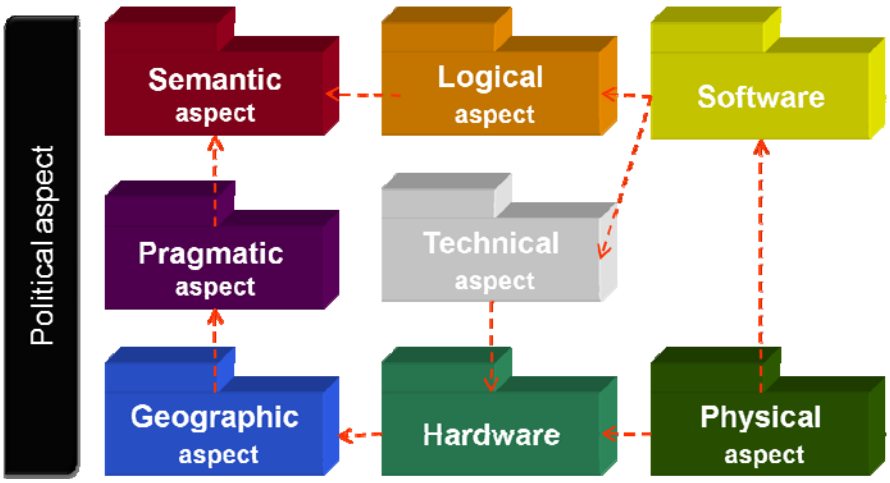
If enterprises are deemed complex systems, how should we address them? Practical questions follow: What is to be represented? How should we deal with the amount of information to be collected and the decisions to be made? These questions call for a methodological framework. At the heart of the software engineering tradition, the “Separation of concerns” principle sets the stage. Over the decades, there have been a few proposed frameworks, varying from a systematic approach (Zachman’s framework with its 30 cells) to simpler and more popular forms (e.g., TOGAF with only four types of architecture). A methodological framework always conveys strong assumptions and expresses an in-depth mindset. These assumptions and mindsets determine the way we see things and the way we act. Therefore, it is of paramount importance to unveil these largely unconscious ideas.

As a methodological framework, the Praxeme method proposes the Enterprise System Topology[2]. It stems from the necessity of capturing all knowledge related to the enterprise, in an actionable manner. “Topology” as a term is to be

[2] At first glance, the term “topology” can be understood in its basic meaning, that given by the etymology: the discourse (*logos*) on the location (*topos*). Using this view, the topology of enterprise system explains how to position elements of information and decision, which appear all the way along the enterprise transformation chain. However, topology also deals with the relations between elements. Although no mathematical approach of the Enterprise System Topology has been attempted yet, there is a striking parallelism between this empirical approach and the mathematical theory. Indeed, the notion of neighborhood obviously applies to the elements of models. It is possible to define a topology for each aspect of the Enterprise System. In reverse, each aspect requires a specific topology with a dedicated notion of neighborhood that takes into account the meaning of the relations between elements. For instance, the valid relations between logical constituents clearly differ from the relations used to express the business knowledge in the semantic model. How far this difference goes is of importance from both a methodological and practical perspective. Obviously, the UML notion of a package is tantamount to the mathematical notion of a subset and it makes sense to ask whether a package is “opened” or “closed”, depending on the topological rules that constrain the design. It is a way of assessing the quality of an architecture.

understood according to its strict etymology: the discourse about the places – answering the question: where should we put every bit of information and decision in the enterprise transformation process? The Enterprise System Topology identifies and links the aspects of the enterprise together (see figure 1).

Figure 1. The Enterprise System Topology



The comprehensive framework identifies and articulates nine aspects. We can formally model each of these aspects, in order to master information and decision-making regarding the enterprise. The "political" aspect is better named "teleonomic"; it gathers scoping elements (elements of knowledge and management): objectives, requirements, vocabularies, rules… These elements are then linked to model elements dispatched in the other aspects, depending on their nature[3].

[3] To follow on from the previous note, it is possible to consider the enterprise itself as a topological space. As a methodological framework, the Enterprise System Topology summarizes relations that exist between elements of various aspects, enabling us to outline a multi-aspect topology. In so doing, we provide the methodology with a mathematical basis for the sake of the analysis and assessment, as far as derivation rules and traceability are concerned. These rules, which automatically link elements from one aspect to another, can be seen as applications – most often injections – and some could reveal themselves as homeomorphisms. An example is the user interface, genuinely derived from a real semantic model by the method. Quality evaluation also benefits from this mathematical tool, since the absence of such formal homeomorphisms indicates divergence and lack of alignment.

This article emphasizes some points related to the framework and the paradigm shift it embodies. Compared to other frameworks, the characteristics of the Enterprise System Topology include:

- insistence on the semantic aspect, which is necessary for establishing a proper representation of the core business knowledge, ahead of the processes;
- place of the logical aspect as an intermediary between business and IT;
- inclusion of the information or data point of view, all the way along the chain of aspects;
- emphasis on relations between the modeling elements (cf. metamodel).

Praxeme recognizes its debt to Zachman's framework, which has inspired the Enterprise System Topology. The latter proposes a simpler – and so, more actionable – order than the former. As regards TOGAF and the frameworks of this generation, we believe, on the one hand, that four or so planes are not enough for organizing the material we have to cope with. On the other hand, as Praxeme focuses on modeling techniques, it is orthogonal to the processes recommended by these repositories of practices. As a result, it is easy to combine the processes and practices with the modeling techniques[4].

How to describe the "business" reality

We use the term "business" as opposed to "IT", meaning the part of the business reality without its software equipment.

As far as business is concerned, we generally use process representations, capability models, use-case models or any other expression describing the business activity. This spontaneous approach of business reality ranks among the functionalist approaches.

It entails a difficulty: we are considering the enterprise in its organizational aspect.

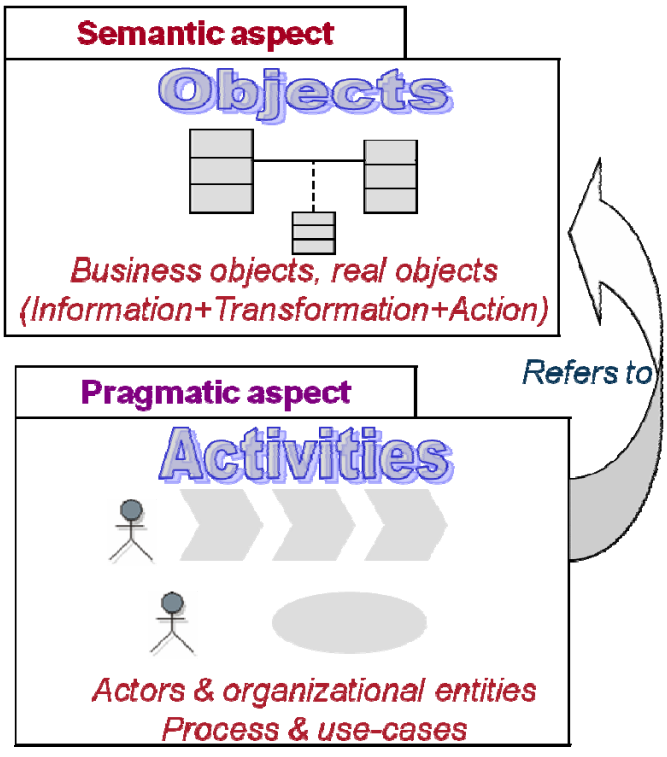
[4] An in-depth comparison of the methodological frameworks would cast much light on our practices and backgrounds. Such a hygienic exercise pertains to methodology, strictly speaking (i.e., an application of epistemology). We definitely need this kind of endeavor if we are to fix our dysfunctional behaviors. This analysis would require more space and is out of the scope of this paper.

Yet, what we see in this aspect are actors and roles, activities and habits, processes and procedures, use-cases... All of these convey organizational choices. Therefore, representations of this aspect can hardly be shared and generalized. When the purpose is convergence, simplification, agility... we need a more generic representation. We need to isolate the core business knowledge, using abstraction and expelling variability. Above this “pragmatic” (organizational) aspect we must recognize a more abstract one, made of business objects, regardless of organizational habits and, of course, independent of technical choices. We call this the “semantic” aspect. The semantic model is not only a sort of conceptual data model; it intends to express the business knowledge. We can use here an object-oriented approach, which provides us with all the tools we need:

- class diagram to structure the concepts,
- state machines to catch the transformation and object life cycles,
- etc.

An object-oriented approach is connoted software but is built upon philosophical works. That explains its ability to efficiently structure representations. It can really empower the formal expression of business knowledge.

Figure 2. The right description of the business encompasses the core business knowledge as well as the processes and organization



How to design the IT system

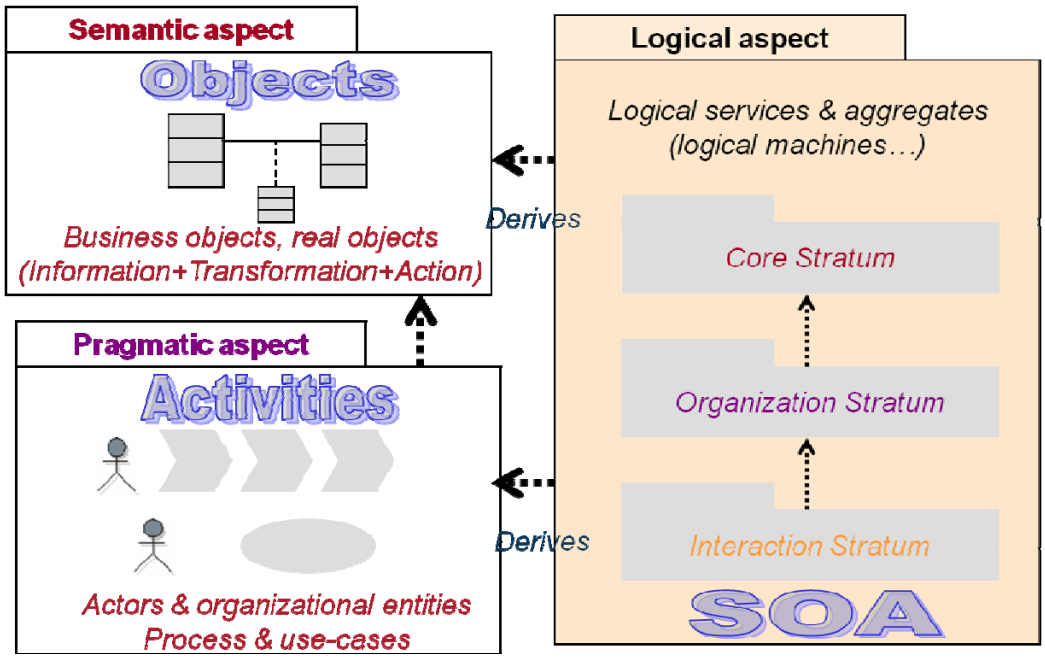
When equipped with the two business representations – semantic and pragmatic – we can search for a better structure for the software solution.

If we conceive this structure directly in terms of technology and technical choices, we will get a representation which will be subjected to technical change. Also, there will be a risk of entering into excruciating details. Such a representation will make it impossible to drive the IS transformation in the long term.

For all these reasons, our framework introduces an intermediate aspect, between business and IT: the "logical" aspect. It is where the structural decisions regarding the software system will be made. For instance, SOA (service oriented architecture) as a style for designing an IT system pertains to the logical aspect.

The logical aspect is linked with the previous aspects. The methodology states the derivation rules which help discover the logical services.

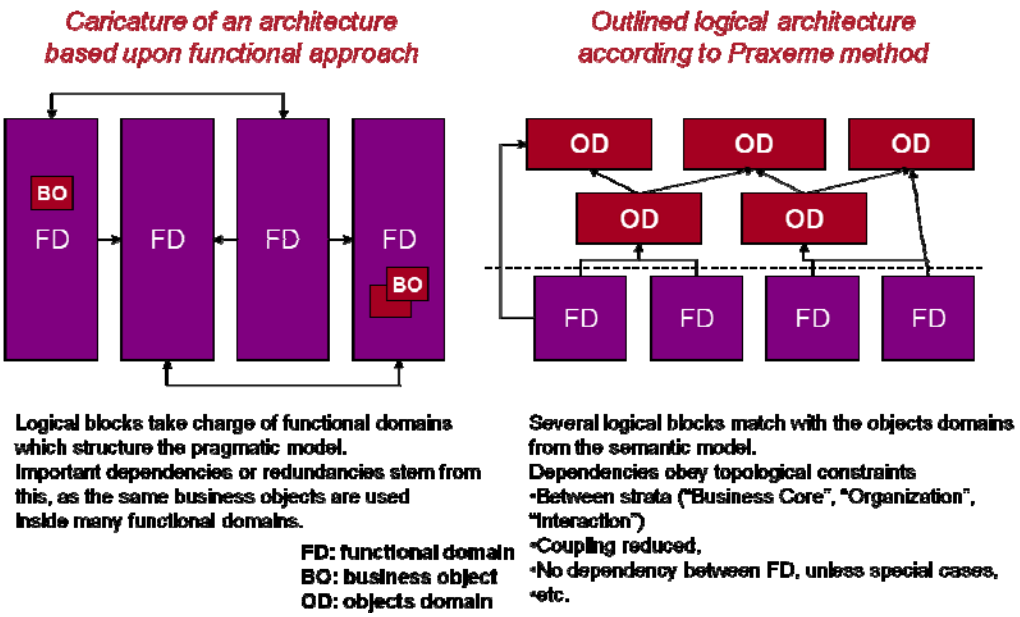
Figure 3. The optimal structure of the IT system takes heed of both models of the business reality



Impact of this approach on a single system

By applying this approach, we deeply change the structure of the system. Indeed, the logical architect receives a list of object domains from the semantic model. Object domains are an alternative way to structure a model, as opposed to functional domains. For more details, see the "Guide to the logical aspect".

Figure 4. The impact of this approach on the architecture of the IT system



The Convergence Approach

How can we master the complexity of the extended enterprise? That is the purpose of the convergence approach, based upon the Enterprise System Topology. This approach provides transformation programs with a strong and willful vision. It helps to prioritize the investments and to drive the transformation, keeping the focus on the essential and avoiding common pitfalls. The convergence goal arises each time a group wants to better integrate its components, either in the case of a merger or in the search for savings.
The resulting process includes these steps (which are more architecture principles than project phases):

1. Separate the concerns
2. Share the core business knowledge
3. Factorize the practices

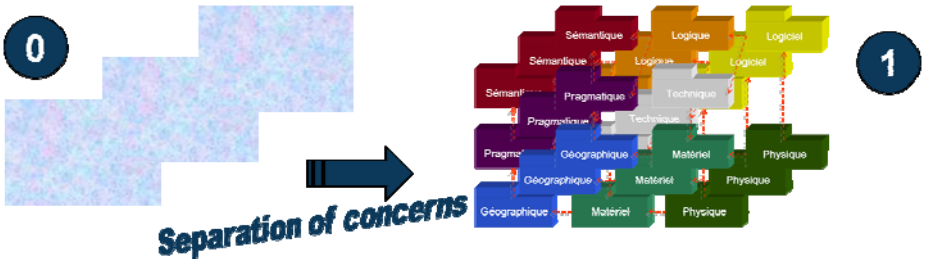
A constant endeavor to isolate the variation points affects every step.

Step #1: Separate the concerns

The starting point is represented by legacy systems where the aspects have not been separated. The systems have been developed, one application after the other. So, the result is just normal. In such systems, it is very difficult to isolate the business rules, to adapt the software to a new organization and to avoid redundancy. The first thing to do is to separate the aspects. The IT system does

not necessarily change at this stage, but at least we obtain a better representation of it and we can compare it to other systems. Figure 5 symbolizes three systems, first as a mess of various concerns and then at the stage where the concerns have been disentangled, in accordance with the Enterprise System Topology.

Figure 5. Adopt the “separation of concerns” principle



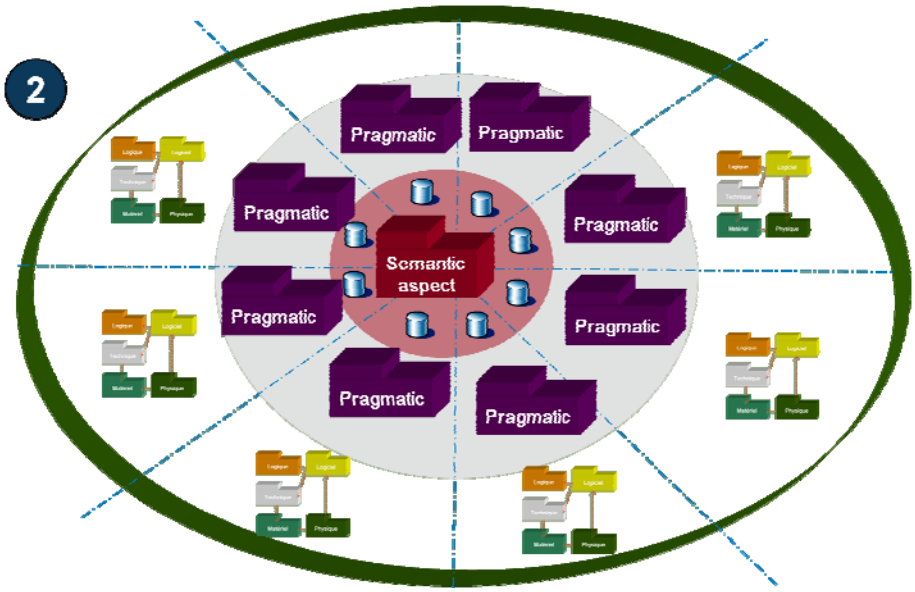
Step #2: Share the core business knowledge

This comparison reveals that the core business knowledge is much the same from one company to another. With good will and appropriate modeling techniques, it is possible to establish a common semantic model. MDM (master data management) and BRMS (business rules management system) are solutions that facilitate the agreement on a common semantic model, by providing means for capturing part of the variations. Therefore, the companies can refer to the same model and adapt it to their context.

For example, national rules that constrain products or practices will be expressed in a BRMS rather than exposed in the common model. In fact, BRMS and MDM are technical mechanisms that are to be considered when it comes to IT architecture. We evoke these mechanisms at this stage of semantic modeling, just to draw attention to the fact that the model may be parameterized and contain meta-data. This remark calls for a specific procedure of modeling.

In the picture below (figure 6), every part of the pie chart represents a different company. At this stage, the building of the IT systems still differs from one company to another, but it refers to a common model as far as core business knowledge is concerned. In addition, there is no attempt at this stage to establish a common representation of business activities.

Figure 6. A common semantic model shared by the entities of the federation



Step #3: Factorize the practices

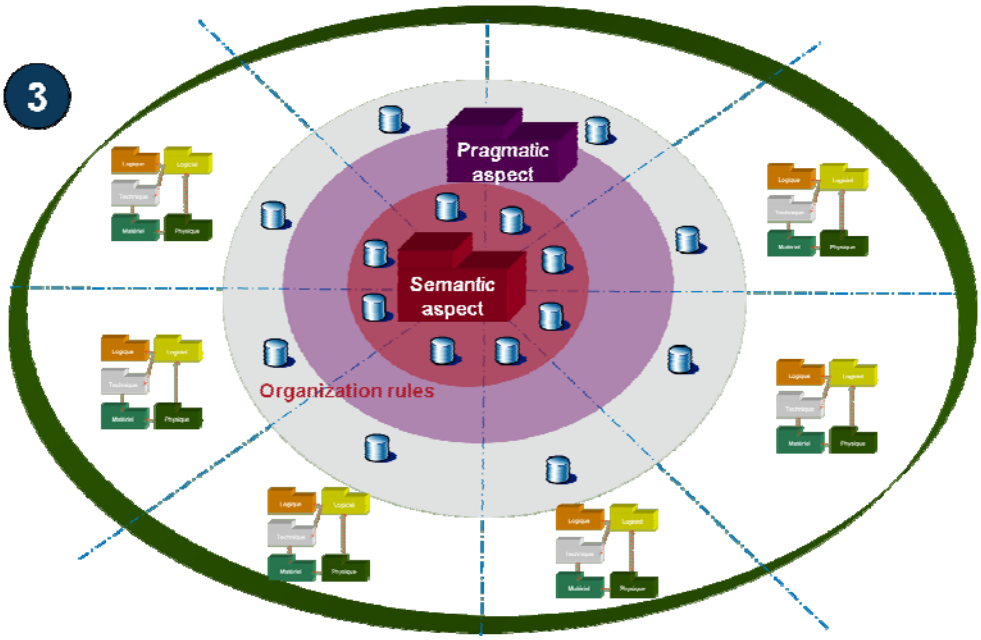
The assumption that characterizes stage 3 states that it is possible to give a generic description of the processes and activities, providing that:

- The appropriate modeling techniques are applied, especially by referring to the semantic model and the lifecycles of the business objects.
- The modeler thinks of various possible usages and prepares the system adaptability by means of parameters.

Regarding the pragmatic aspect, the solutions provided in the field of business process management are particularly useful. These solutions will be implemented later on, on the technical architecture and in the software. Knowing that, the modeler is free for a more generic design[5].

[5] To go back to our discussion on mathematical formalization, we can now suggest treating several companies as “adjunction spaces”. Interoperability and convergence can be approached through sets of components – in the latter case – or sets of flows – in the former one. Both goals – interoperability and convergence – have to be specified in terms of the aspects that are targeted. For instance, interoperability can be sought at the technical level only (without convergence of the content) or at the software level (implying identity inside the antecedent aspects). The corresponding transformations can be represented in terms of applications and their combinations (adjunction, disjoint union, product…). Intuitively, this is a potential starting point for establishing a rigorous approach of enterprise systems, including federations of systems and their evolutions. Firstly, such an approach would benefit from findings in the field of architecture and complexity measurement (cf. Y. Caseau and D. Krobs, M. Aiguier). Secondly, it would expand to multi-

Figure 7. A single description of processes with their parameters



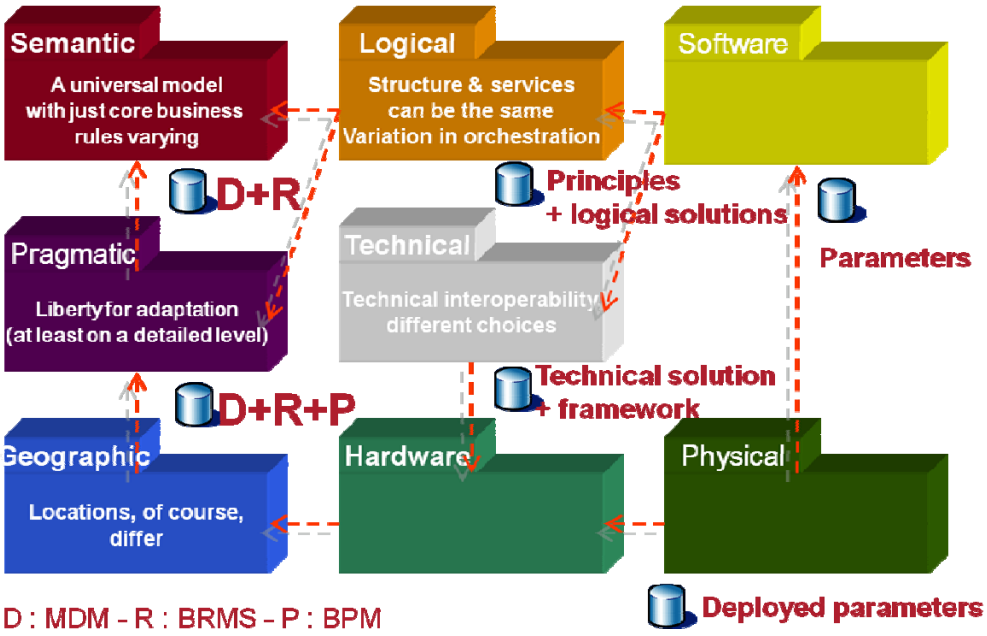
Operating rule: “Isolate the variation points”

The sequel can go further. Each time, for every aspect, the idea is to seek the possibility of sharing a common description and expelling the causes of variation in the shape of an ad hoc device. The technical choices are not the only ones that can be shared, nor the most critical, nor necessarily the priorities. One has to ask where is there the most value in convergence?

For instance, a logical model with specification of services in an SOA approach is easier to consider as a reference rather than its translation into software. Indeed, the software takes into account various technical architectures…

systems. As it includes the multi-aspect dimension, a consequence of the “separation of concerns” principle, it does not limit itself to a mere evaluation tool but also conveys strong recommendations for transforming the systems.

Figure 8. Types of variation points depending on the aspects



MDM = Master Data Management; BRMS = Business Rules Management System; BPM = Business Process Management

Conclusion

The convergence approach summarizes a kind of “utopian” approach, since it reverses the approach generally adopted. It is typical of a top-down approach, which we observe more and more scarcely. Indeed, we are facing a paradox: on the one hand, decision-makers are calling for innovation and transformation more and more often; on the other hand, the practices of design and architecture have dramatically regressed and are receding in front of the so-called pragmatic approach. The complexity of the matters is put forward as an alibi for avoiding tough decisions. As a result, it has become an urgent matter to propose concrete guidelines and to reform our practices. In the face of complexity, Praxeme is an attempt to reactivate the methodological tradition and to provide proper guidelines.

This method and the multisystems approach have been applied to several programs in various sectors: UAV[6] control systems (SAGEM), insurance (Azur-GMF, SMABTP, AXA Group)… *Sustainable IT architecture* illustrates the methodology that has been applied to the overhauling of an entire information system. The initiative for an open method has already made guides and training

[6] Unmaned Air Vehicle.

support available. We are aware the current corpus lacks many topics and procedures. Let us end this article with a call for contributions to the initiative and to build the method our enterprises definitely need.

Bibliography

Caseau Y., Krob D. et Peyronnet S., *Complexité des systèmes d'information : une famille de mesures de la complexité scalaire d'un schéma d'architecture*, Génie logiciel 2007

Aiguier M., Le Gall P., Mabrouki M., *A formal denotation of complex systems: how to use algebraic refinement to deal with complexity of systems*, Technical Reports available on www.epigenomique.genopole.fr

Strategy & Leadership in turbulent times, McKinsey Quarterly 2010

Aiguier M., Le Gall P., Mabrouki M., *Complex software systems : Formalization and Applications*, International Journal on Advances in Software , 2(1):47-62, 2009.

Bonnet P., Detavernier J.-M., Vauquier D., *Sustainable IT Architecture*, Wiley, 2009.

Vauquier D., *Praxeme methodological guides*, www.praxeme.org

www.enterprisetransformationmanifesto.org ("In the face of complexity, this manifesto articulates core principles and offers an escape from confusion, gloom and doom. It aims to reinforce our ability to act.")

Curley M., *Managing Information Technology for business value*, Intel Press 2007

Depecker L. rédacteur, *Terminologie et sciences de l'information*, Le savoir des mots 2006

Rozanski N., Woods E., *Software Systems Architecture*, Addison Wesley 2005

Longépé C., *Urbanisation du Système d'information*, Dunod 2001

C. Hofmeister, R. Nord, D. Soni, *Applied Software Architecture*, Addison Wesley 1999

Meinadier J.P., *Ingénierie et intégration des systèmes*, Hermès, 1998

Strassmann P. A., *The politics of information management*, The Information Economics Press 1995

Vauquier D., *Développement Orienté Objet*, Eyrolles, 1993

H. Tardieu, A. Rochfeld, R. Coletti, *La méthode Merise*, Editions de l'Organisation, 1989

Léo Apostel, *Syntaxe, Sémantique et Pragmatique*, in *Logique et Connaissance Scientifique* dir. Jean Piaget, coll. La Pléiade, Gallimard, 1967

Jean Piaget, *Épistémologie des mathématiques*, in *Logique et Connaissance Scientifique* dir. Jean Piaget, coll. La Pléiade, Gallimard, 1967