MATCHING ERP FUNCTIONALITIES WITH THE LOGISTIC REQUIREMENTS OF FRENCH RAILWAYS

A Similarity Approach

Iyad Zoukar , Camille Salinesi
Centre de Recherche en Informatique, Université Paris 1 – Panthéon – Sorbonne
90, rue de Tolbiac 75013 Paris – France

Keywords: ERP, Business Process, Requirements Matching, Meta-Modelling, Similarity Analysis, Requirements Elicitation.

Abstract: Ensuring the adequacy of Enterprise Resource Planning (ERP) implementations to business requirements is still an issue that needs to be addressed. One important cause of inadequacy results from the lack of attention paid to the precise and systematic analysis of how well ERP functionalities match the business requirements. One well known reason for this is that the language used to define ERPs is different from the one used to define business requirements, and there is no technique available so far to evaluate systematically similarities between ERP functionality models and business requirements models. Our approach to this issue is (i) to materialise with a unified goal/strategy modelling language both the ERP functionalities and the business requirements, and (ii) to systematically specify using a similarity model how a given ERP functionality model and a business requirement model should match together. This paper outlines this matching method and explains how the similarity model was developed in a systematic way.

1 INTRODUCTION

Ensuring the adequacy between an organisation's requirements and the Information System (IS) functionalities represents a major issue of any IS project. This especially holds with ERP systems in which functionalities are already designed and built-in for standard business processes. Our work at SNCF (the French railways company) consists in developing a methodology that would help specify how the functionalities provided by the PeopleSoft ERP match the requirements of the supply chain processes that it shall support.

As often in this kind of project, the approach initially taken in the project was mostly driven by PeopleSoft functionalities. However, these are only documented at a very detailed level transactions, operations and data structures, etc), whereas at SNCF stakeholders think in terms of their goals, tasks and outcome. This results in a difficulty for stakeholders to adapt to the language of ERP experts, and furthermore, in a difficulty to foresee how the system will fit to their requirements. This mismatch exposes the project to a classical and well-documented danger of failure (Standish, 1995) (Davenport, 2000).

Our approach was that the relationship between the system and the business should be materialised using a unique language that gathers the business and the system perspectives (Salinesi, 2003). The formalism we use, called Map (Rolland, 2001), is built on two central concepts that are natural to business experts, goals and strategies: *goals* are used to identify the business processes for which the system provides (at least partial) support; *strategies* indicate how the business intends to achieve goals.

Our purpose at SNCF was to establish a way of working to align the business requirements and system functionalities. To do that, we adapted the general IS evolution framework (Salinesi, 2003). The framework that we developed clearly shows that an important aspect of matching is the production of statements about the similarity between the ERP and the business requirements. A typology of similarity predicates was thus developed.

The next section introduces the matching framework. In section 3 the similarity topology is presented. Examples from the SNCF project are given in section 4. Related works and conclusions are respectively presented in sections 5 and 6.

Zoukar I. and Salinesi C. (2004).
MATCHING ERP FUNCTIONALITIES WITH THE LOGISTIC REQUIREMENTS OF FRENCH RAILWAYS - A Similarity Approach.
In *Proceedings of the Sixth International Conference on Enterprise Information Systems*, pages 444-450
DOI: 10.5220/0002628404440450

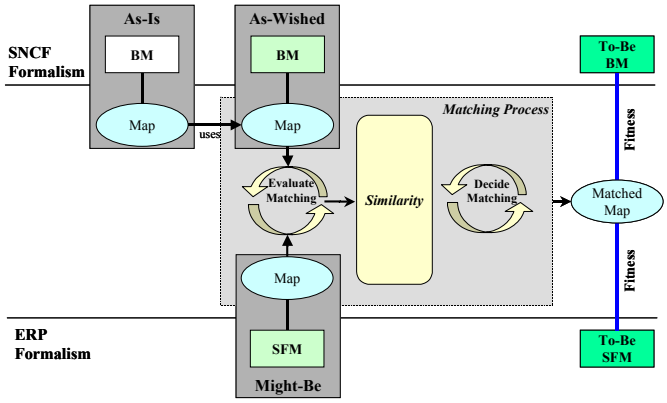

Figure 1: Adapting the Evolution Methodological Framework to the context of the ERP project at SNCF

2 ERP MATCHING FRAMEWORK

As proposed by (Jarke, 1993), a general framework was defined to situate the different concepts that are needed in a method dealing with IS evolution. Besides, there are different contexts of IS evolution; each context can be characterised with a more specific framework (Salinesi, 2003).

One of the evolution contexts concerns IS customisation from a product family. This context is met in ERP implementation which corresponds to the SNCF project nature. In the generic '*customisation from a product family*' framework, four families of models have to be managed. The requirements of the organisation are expressed in the *As-Wished BM (Business Model)*. The *Might-Be SFM (System Functionality Model)* reflects the functional capability of the product family. The *To-Be BM* and its counterpart, the *To-Be SFM* result from a *matching process* which searches for the best fit between the organisational requirements (expressed in *As-Wished BM*) and what is proposed by the ERP, (*Might-Be SFM*).

We adapted this generic framework in order to take into account all the kinds of models actually used in the SNCF project. As shown in figure 1, numerous kinds of models were used. These are organised in the four aforementioned families: (1) *As-Wished BM* capture the business processes that the organisation requires for the future. (2) The *Might-Be SFM* represent the different functionalities provided by the ERP. (3) The *To-Be BM* represent the business processes after the project. (4) The *To-Be SFM* specify the ERP after the project, i.e. after parameterisation, re-development of existing functions, and development of new specific functions.

An important goal of the *Matching Process* is to ensure the fitness relationship between the business and the system at the end of the project, i.e. to make sure that the delivered system fits to the use of the future business. On the business level, this calls for adaptation of the Business Processes; on the system functionality level, this calls for customisation of the ERP and of the legacy system.

As illustrated above, the situation at SNCF is a complex one as different formalisms are used in each of the four families of models that does not allow a systematic matching. Besides, the project managers found that the business processes are too complex to be defined at once, and therefore chose to develop them by analysing the current situation. This choice appears through a fifth family of models, namely the *As-Is BM*.

Given this complex situation, our proposal was to use the map formalism to reflect the existing As-Is BM, As-Wished BM and Might-Be SFM in a unique way. Let us notice that, the other way round, the additional work that was needed to develop these maps was also useful as it allowed to synthesise the models at hand in abstract terms and remove cumbersome details. Last, the matching process was facilitated as it resulted in producing *similarities* between specifications expressed with a unique meta-model.

3 THE SIMILARITY TYPOLOGY

The central technique used during the matching of Business Models and System Functionality Models was the analysis of similarities between them. Intuitively, a similarity (represented with the "≡" symbol in Figure 2) expresses a resemblance between elements from two different models. Our assumption is that similarities between maps can be expressed with predicates that can be listed under the form of a structured typology.

3.1 Towards a generic similarity typology

Obviously a relevant map similarity typology could have been defined in an ad-hoc sort of manner. However, besides the fact that this might be error prone, the resulting typology would be dependent on the specific requirements specification formalism used in this paper, i.e. the map formalism. To overcome these difficulties, we adopted the approach developed in (Rolland, 2003) (Etien, 2003) to specify a typology of gaps between maps. First, we searched for a generic similarity typology, i.e. a typology that is independent of the formalism used to express the As-Wished and the Might-Be models. Then, the map similarity typology was developed as an instance of the generic similarity typology. The instantiation was guided by an accurate definition of the map models in terms consistent with those of the generic similarity typology. Using this approach, other similarity typologies such as a similarity typology between object oriented models or a Use Case could be easily generated too.

As shown in figure 2, a generic level is first used to abstract the specific meta models used in ERP projects. The left part of figure 2 shows that the generic meta-model is instantiated by specific meta-models, and that specific similarity typologies are issued from the generic similarity typology. The purpose of the former instantiation is to identify the key elements and structures of the specific meta-model. The definition of the specific similarity typology is made systematic by the knowledge of the link between the similarity typology and meta-model at the generic level, combined with the instantiation relationship between the generic meta-model and the specific meta-model. A description of the generic Meta-Model can be found in (Etien, 2003).

3.2 The generic similarity typology

Any meta model is composed of elements with properties. Besides, the structure of meta models is shown though element composition and through links between elements. Based on this, the generic typology of similarity predicates emphasises that given a pair of elements, (i) their properties can be similar, and (ii) their structure can be similar. As the right part of figure 2 shows, there are thus two classes of similarities, intrinsic similarity and structural similarity.

A pair of elements has an *intrinsic* similarity if they have similar properties. Element properties can be considered similar if they have a close semantics. In the first place, intrinsic similarity relates to synonymy. However, it can also be about the hyponymy (or the other way round hyperonymy) relationship.

The *structural* similarity deals with the composition of elements and their organisation within models. There are thus two classes of structural similarity: compositional similarity, and relational similarity. Contrary to intrinsic similarity that involves only the two compared elements, the structural similarities imply comparisons between other elements that are related to the two compared ones.

Besides, as shown in figure 2 by the aggregation link from structural similarity class to the similarity class, a structural similarity is a complex one and involves other similarities. For example two elements have the "same components in a composition" if each component in one element has a semantically "same" counterpart in the composition of the other element.

There are therefore four main similarity classes. These are defined as follows:

(i) *Synonymy* is a relationship between two elements that have either properties or types with a similar or close meaning. There are two kinds of synonymy:

- Two elements have a synonym type if their types are equal or have a common super-type (they are then cousins).
- There are different degrees of resemblance possible between the properties of a pair of elements: two elements have the *same property* when their properties have exactly the same name and the same

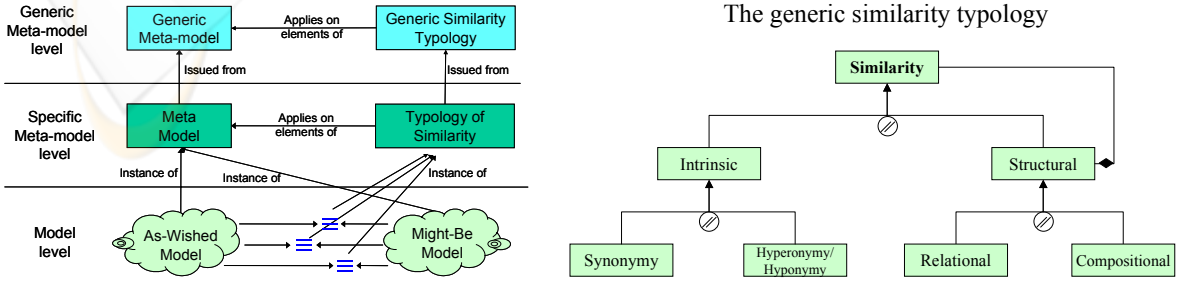

Figure 2: Overview of the similarity elicitation approach

meaning (for example an actor in a Use Case model and an actor in a sequence model); they have *alike properties* when their properties are identified with different words but have the same meaning (for example two classes that specify the same business object in two different ERP modules); or they have a *resembling property* when the properties have different names and values, but they still have a close meaning (like for example a standard business object in two different ERPs).

(ii) *Hyponymy/Hyperonymy* relates two elements when the meaning of the one subsumes/is subsumed by the meaning of the other. As with synonymy, hyponymy/hyperonymy similarity can be defined on the type and on the properties of elements:

- with respect to type, hyponymy/hyperonymy comes down to a *father* or a *son* relationship between the types of the involved elements. This is for instance the case of a UML class in a model that appears as an abstract class in another model.
- With respect to properties, two elements are in a hyponymy/hyperonymy relationship if the properties of the ones *includes/extends* the properties of the other. This is for instance the case when the attributes of one class are included in the collection of attributes of another class.

(iii) *Relational* similarities are defined between link elements that are connected to similar source/targets, or between elements that are related to the rest of their models through similar links. As table 1 shows, there are different kinds of relational structure similarity predicates.

These include (without being restricted to): *same number of links* (when two elements are source/target of the same number of links), *same number of links entering in a node* (when two elements are source of the same number of links), *same number of links outgoing from a node* (idem, the other way round), *same/alike/resembling source, target, or source and target* (when two links have similar extremities), *same depth* (same max distance between nodes and leaves of the trees they belong to) or *same height* (same max distance between nodes and the root of the trees they belong to).

(iv) *Compositional* similarities deal with compound elements that are similar in their composition, and with elements that belong to similar compositions. Table 1 quotes a number of compositional structure similarity predicates: *same cardinality of a component* (when two compound elements have the same number of components), *same / alike / resembling components in a composition* (when the compositions of two compound elements are comparable), *same / alike / resembling common component in a composition* (when part of the compositions are comparable), *part of same / alike / resembling compound* (when an element is similar to a component of another element).

4 EXAMPLES OF APPLICATION AT SNCF

Three sets of maps were produced as required in the framework presented in section 2: As-Is, As-Wished and Might-Be maps respectively representing the current BP at SNCF, the wished BP at SNCF, and the BP supported by PeopleSoft. This section reports examples of use of similarity typology to match those maps in order to produce the matched maps that represent the To-Be, i.e. the business and the system situations after the ERP implementation.

In figure 3 the two extracts of maps represent the organisation requirements (on the right) and what is proposed by PeopleSoft (on the left). A number of similarities can be specified using the typology of similarity predicates. For example, there are intrinsic similarities between the As-Wished and Might-Be maps. Indeed, synonymies with equal types and equal properties can be defined between the pair of intentions *Build production plan* and *Solve production plan* and between the two strategies *For items managed with the forecast* and *For items managed with orders*. Although their statement looks very different at first glance, there is a hyponym/hyperonym similarity between the

Table 1: The generic similarity predicates

Synonymy	*Hyperonymy Hyponymy*	*Relational*	*Compositional*
Type	Type	Same number of links	Same cardinality of a component
Equal type	Father type	Same links number entering in a node	Same components in a composition
Cousin type	Son type	Same links number outgoing from a node	Alike components in a composition
		Same source	Resembling components in a composition
Property	Property	Alike source	Same common component in a composition
Same property	Includes property	Resembling source	Alike common component in a composition
Alike property	Extends property	Same target	Resembling common components in a composition
Resembling property		Alike target	Part of same compound
		Resembling target	Part of alike compound
		Same source & target	Part of resembling compound
		Alike source & target	
		Resembling source & target	
		Same depth	
		Same height	

By frozen fence and the *Based on the first two months of the planning horizon* strategies as the former proposes a more general solution with more options than the latter. Besides, looking at the relational similarities between the two map extracts, it was found that they have the same depth. This indicates that they are described at the same level of abstraction. A closer analysis of the bundles composed of the pair of goals and the strategies between them shows that there are however compositional similarities. As one could expect from the different number of strategies, the two bundles do not have the same components. However, the bundles have an alike common components similarity as two of the strategies in the As-wished bundle have an equal in the Might-Be, and one has a hyperonym in the Might Be. No similarity could be found by looking at the height with respect to the refinement links (these links are not shown in the figure for the sake of space).

This experience showed us that a number of alternative To-Be solutions could always be envisaged. This calls for a systematic way to explore alternatives, i.e. to identify them and to find the best To-Be solution. Map similarity typology facilitates the identification and the construction of the different matching alternatives. Because similarity predicates are generic, automation of similarities detection is possible using our generic typology. In real project size, the number of As-Wished and Might-Be models can by very great and manual similarities detection is impossible that why CASE tools are needed.

Similarities discovered using the generic typology present an important element of a decision making process during the matching process to choose the best solution for the organisation.

Figure 4 shows an example of situation in which alternative matchings can be explored. In this situation a number of wished business strategies and system functionalities were identified to '*Solve production plan*'. These are respectively shown by the right and left map extracts in Figure 3. Matching these maps using the above leads to at least three possible To-Be maps as Figure 4 shows it: Alternative A1 results from an *As-Wished driven matching*, i.e. the solution is mostly similar to the As-Wished BM. Alternative A2 is issued by a *Might-be driven matching*. This solution adopts all the strategies from the Might-Be SFM that can be useful to SNCF. Alternative A3 can be surfaced by a *Two-way matching*.

Discussions were raised for choosing among the alternatives resulting from the matching process. Decisions could be made using a number of criteria such as the percentage of specific development in the ERP, non-regression, cost, delay, available competencies, negotiation margins, etc.

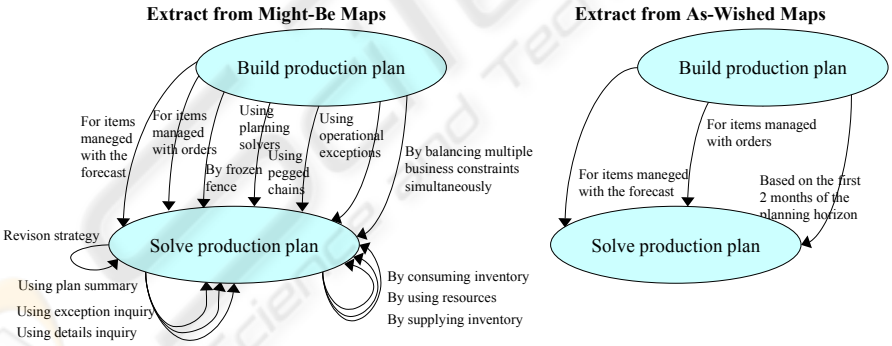

Figure 3: Two extracts from As-Wished and Might-Be maps

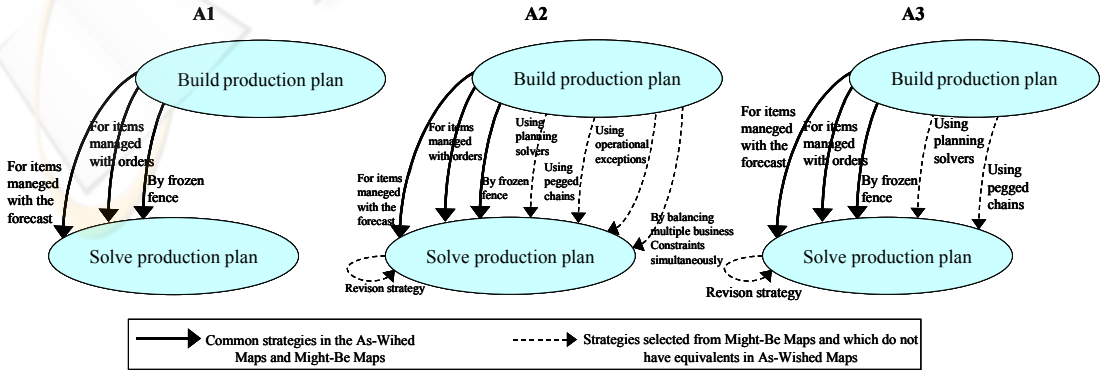

Figure 4: Alternative matching possibilities

5 RELATED WORKS

Despite the widespread adoption of ERPs by business organizations (Joseph, 1998), there is still a lot of academic research need on ERPs from the engineering perspective (Borell, 2000).
Two families of approaches were developed to address the matching issue: the management approaches and the system approaches. In the management family, research intends to define the impact of ERP installation on corporate culture (Krumbholz, 2000), organization (Robey, 2002), or business processes (Esteves, 2002). In the system family, the purpose has been to guide the identification and selection of the most appropriate ERP (Ncube, 2000), and to elicit requirements to inform the most adequate customisation of ERPs (Finkelstein, 2002).
Our approach is in-between the two families. Its main assumption is that in an ERP project, the issue of organizational change and system engineering are intertwined. Therefore, their matching should be neither driven by the business nor driven by the system functionality, but both.
Several works have already been achieved on similarity measure. For example, (Castano, 1993) proposes to evaluate components reusability through conceptual schema. (Jilani, 1997) used similarity measures to select best-fit components. Similarity metrics for heterogeneous database schema analysis were introduced by (Bianco, 1999). Our similarity approach is inspired by Castano and Bianco.

6 CONCLUSION

Our experience in an ERP project at SNCF confirmed to us the importance of the matching between the world of business and the one of systems and the necessity of using similarity analysis techniques to support this matching. Our approach to this issue was to use a goal/strategy model called map as an in-between language on which the matching process can be achieved in an efficient way. So far, we have: (i) developed a specific methodological framework that sets in context the business models, system functionality models, and the matching approach, (ii) defined the issues of matching business models and system functionality models and (iii) adopted a similarity typology to systematise the specification of the result from matching activities.
The next tasks in our research program are: to further document our methodological framework, to complete the similarity typology and to develop a completely guided methodological process model.

REFERENCES

Bianco G. and al. A Markov Random Field Approach for Querying and Reconciling Heterogeneous Databases. *DEXA'99*, Florence, 1999.

Borell A. Hedman J. CVA Based Framework for ERP Requirement Specification. *IRIS 23. Laboratorium for Interaction Technology*, University of Trollhattan Uddevalla, 2000.

Castano S. De Antonellis V. A Constructive Approach to Reuse of Conceptual Components. *2d International Workshop on Software Reusability*., Italy, 1993.

Davenport T.H. Mission Critical: Realizing the Promise of Enterprise Systems. *Harvard Business. School Press*, MA, 2000.

Esteves J. Pastor J. Casanovas J. Monitoring Business Process Redesign in ERP Implementation Projects. *Americas Conference on IS*, Dallas, 2002.

Etien A. Salinesi C. Towards a Systematic Definition of Requirements for Software Evolution: A Case-study Driven Investigation, *EMMSAD Workshop in the CAISE Conference*, Austria, 2003.

Finkelstein A. Bush D. Requirements Elicitation for Complex Safety Related Systems. *London Communication Symposium*. London, 2002

Jarke M. Pohl K. Establishing visions in context: toward a model of requirements process. *12th International Conference Information System*. Orlando, 1993.

Jilani L.L. Mili R. A. Mili. Approximate Component Retrieval: An Academic Exercise or a Practical Concern?. *WISR8*, Columbus, Ohio, 1997.

Joseph T. Swanson E.B. The Package Alternative in Systems Replacement: Evidence for Innovation Convergence, in Information Systems Innovation and Diffusion. *Larsen-McGuire (eds.)*, 1998.

Krumbholz M. Maiden N.. How Culture might impact on the Implementation of Enterprise Resource Planning Packages. *CAISE'00, Springer Verlag*, 2000.

Ncube C. Maiden N. COTS Software Selection: The Need to Make Tradeoffs Between System Requirements, Architectures and COTS/Components. *COTS'00 Workshop for ACSE00*, Ireland, 2000.

Robey D. Ross J.W. Boudreau M-C. Learning to Implement Enterprise Systems: An Exploratory Study of the Dialectics of Change. *Journal of Management Information Systems*, 2002

Rolland C. Prakash N. Matching ERP System Functionality to Customer Requirements. *5th IEEE Requirements Engineering*, Toronto, 2001.

Rolland C. Salinesi C. Etien A. Eliciting Gaps in Requirements Change. *Requirements Engineering Journal*, 2003.

Salinesi C. Rolland C. Fitting Business Models to Software Functionality: Exploring the Fitness Relationship. *CAISE'03*, Springer Verlag (pub), 2003.

Standish. Standish Group. Chaos. www.standishgroup.com /sample_research/chaos_1994_1.php, 1995.