Comparison of Business Process Models as Part of BPR Projects

Mouna Tkaa,*, Sonia Ayachi Ghannouchiab

aNational School of Computer Science, RIADI Laboratory
Manouba University, Manouba 2010, Tunisia
bHigher Institute of Management, Abdlaaziz il Behi Street, Sousse 763, 4000, Tunisia

Abstract

Business Process Reengineering (BPR) is one of the existing methods in the literature that lead to evolutionary changes and adjustments that have become a necessity in today’s business. As part of BPR projects, we need to compare process models to reference models to detect differences and propose improvements thereafter to remedy them. This paper presents a state of the art on works that were interested in comparing models of business process models and defines an approach for comparing two business processes and measuring the gap between them. The proposed approach has been developed and was tested with software development processes.

© 2012 Published by Elsevier Ltd. Selection and/or peer review under responsibility of CENTERIS/SCIKA - Association for Promotion and Dissemination of Scientific Knowledge Open access under CC BY-NC-ND license.

Keywords: business process model; comparison of models; reference model.

1. Introduction

Under the influence of competition and technological change, the successful company must not only produce and distribute services but also create value and grow. To grow, it must change. The company is seen

* Corresponding author. Tel.: +216 97851222
E-mail address: mouna.tka@gmail.com.
as an organization composed of several business processes; therefore the change has to begin from these processes. The business processes are defined by ISO as a "Set of interrelated or interacting activities which transform inputs into output". A business process is a collection of related tasks that lead to a specified goal. Many modeling notations are available to capture business processes, including Event-driven Process Chains (EPC) [1], UML Activity Diagrams [2] and the Business Process Modeling Notation (BPMN) [3].

Many are the methods of improvements in business processes that exist in literature from which we notice the Total Quality Management which is a management of an organization centered on quality and the long-term success through customer satisfaction and providing benefits for members of the organization. It provides a gradual improvement in business processes, but it is very spread out over time.

Another method is benchmarking which is seen as a lever for innovation in the enterprise. It is at the same time an analysis to calibrate the best drawing points, a mindset and a management style. Benchmarking can be done internally between departments, or entities, and externally, with its various suppliers, distributors, competitors, market products. The idea is to compare different objective or subjective criteria, but at least quantifiable rules that can be made to have the same analysis systems between the different members who will perform the benchmark. The goal is to determine the criteria on which we do not need to invest because they are satisfied better than others and those where improvement work is necessary to achieve the optimum.

In addition, we have Business Process Reengineering (BPR) to which we are particularly interested in this paper. BPR is described by Hammer and Champy as "questioning fundamental and radical redesign of organizational processes to achieve dramatic improvements in current performance on cost, service and speed." That means that existing processes are carefully reviewed and redesigned in order to improve and complete more effectively and efficiently the functions of the company. It must be supported by a modeling process because it was increasingly necessary to explicitly define the process. In the context of reengineering business processes, organizations compare business process models to identify operational commonalities and differences. Such comparisons are, for example, necessary when organizations merge and need to determine and resolve the differences between their operations, and when an organization needs to check whether its operations conform to a company-wide or industry-wide standard. This standard or these reference models are models that have proven their effectiveness and success. In such way companies will find improvements to remedy problems or gaps in its business process.

Particularly, the objective of the work presented in this paper is to study the way in which process models can be compared, and thus find a comparison method that will detect the gaps and provide an opportunity to complete a draft reengineering business processes, minimizing waste of resources and time and improving products.

The paper is structured as follows. Section 2 presents works that have focused on the comparison of business process models which inspired us to define an approach to compare two business processes, measure the gap between them and so find ways for improvement. This method will be described in the third section. In section 4, we present a prototype implementing the proposed approach. Next, in section 5 we illustrate our work with a case study. Finally, in section 6 we conclude and refer to future work.

2. Related Work

Our work can be related to three areas of research: measuring similarity between business process models based on behavioral comparison [3], Measuring similarity between semantic business process models [5] and finally measuring similarity between business process models' labels using tools and components to develop matchers that determine matchings between the activities of two business process models, particularly, we focused on the "ICoP" framework: Identification of Correspondences between Process Models [6].
In the context of comparison of business process models a novel approach for measuring the degree of similarity of business process models has been studied. This approach considers linguistic and behavioral aspects of process models to calculate a degree of similarity [7]. It builds on the vector model from information retrieval, an abstract representation of process behavior as causal footprints [8] and an automatic matching of functions across process models. It manipulates the Event-driven Process Chains (EPCs). The EPC is a popular business process modeling language that was introduced in [9]. EPCs offer function type elements to represent these activities, event type elements describing pre- and post-conditions of functions, and three kinds of connector types including AND, OR, and XOR. Control flow arcs are used to link these elements [10].

Beyond that, there are other works on measuring similarity between semantic business process models. In fact, a business process may be modeled in different ways by different modelers even when utilizing the same modeling language. An appropriate method for solving ambiguity issues in process models caused by the use of synonyms, homonyms or different abstraction levels for process element names is the use of ontology based descriptions of process models [5].

A method introduced in [5] describes high-level Petri nets in OWL (Web Ontology Language) and get similarities between Semantic business process models as an instantiation of the Pr/T net ontology [11]. Another interesting work which can be related to our paper is the framework “ICoP”: a re-usable framework for identifying correspondences between activities in one process and equivalent activities in a similar process, while taking into account that equivalent activities may be modeled at different levels of granularity and have different control-flow relations to other activities. The framework is specifically tailored to also deal with complex 1:n matches (i.e. each activity can be matched to an arbitrary number of other activities). The framework consists of a flexible and adaptable architecture and a set of reusable matcher components [6]. Matchers can be developed in the framework by composing them from existing components and, if desired, newly developed ones.

In this paper we adopt some concepts from this area for matching function labels (textual similarity), and we combine this approach with the calculation of semantic similarity taking into account the transitions between activities and the structure of the model.

3. Proposed approach

Our work is only focused on static structure of business process models. For this we studied several existing methods that we outlined in Section 2. They guided us to design a simple method of comparison while trying to overcome the disadvantages already revealed especially complexity limitations. This method will compare a business process model described in BPMN to another reference model, to determine similarity degrees between them and thus allows companies to improve their business process following the results of the comparison within a re-engineering project.

Considering a process model as a set of activities and a set of transitions, comparing two Models means logically comparing the activities’ sets and transitions’ sets. From this idea, we must begin by comparing activities taking into account semantic and syntactic similarities then the comparison of transitions will come after matching activities from both sides. We suppose that we have extracted data (Activities' labels and transitions) from the model and put it on a workable format for example XPDL files derived from BPMN model [12].

The approach is formalized into a five step process whose aim is to support the development of the future applications of business process models’ comparison:

1. The comparison between the numbers of business activities of the two Models:
If the difference in the number of business activities is bigger than a fixed limit, we abandon the comparison. This step is for optimization because there is no need to execute the whole algorithm in that case.

2. The syntactic comparison: This step consists in building a matrix of syntactic distances between different labels of the two Models. At this level we will apply a new algorithm to calculate these distances: the Levenshtein's algorithm [13] improved. Levenshtein distance is defined as the minimum number of elementary operations that are required to transform a string S1 into a string S2. There are three possible transactions: replacing, deleting or adding a character. This measure takes its values in the interval \([0, 1]\). This distance is greater as the difference between the strings is.

This algorithm is used mostly to compare 2 words, or a wrongly spelled word against a dictionary, and for this it does the job very well. Unfortunately in more complex strings, like phrases, complex names or others, this doesn't really work. We couldn't apply just the Levenshtein distance for example to compare "City Food Company" to "Food City Co" because they would be a lot different where in fact it is the exact same entity, written a bit differently.

So here appears the need of an algorithm that takes into consideration the change of order of the words and abbreviations besides the traditional Levenshtein distance. We conceived a class named "StringDiffCompareClass" illustrated in Figure 1 which implements functions used to:

- Compare abbreviations of words in the strings from both sides.
- Compare the words in the strings from both sides reordered in Alphabetical order.
- Calculate the classic Levenshtein distance.
- Make an average of these comparisons according to different weights passed as parameters.

The results of this comparison will be stored in a matrix of \(n\) columns and \(p\) rows where \(n\) is the number of activities of the first business process Model and \(p\) the number of activities of the second business process Model. Thus row \(i\) will contain the distances of the activity of order \(i\) in the second model with all activities in the first one.

3. The semantic comparison: If the labels are not identical, then they will be compared semantically, two labels are equivalent if they are synonyms referring to given dictionary. Several dictionaries and thesauri are free and available [14]. So we conceived the class "SemanticComparer" which implements the function "GetSynonyms" presented in Figure 1. That function, as its name indicates, gets all the possible synonyms of an expression. Then we update the matrix of distances following last results of this comparison.

4. Matching the activities of two models: Based on the matrix of distances between activities, we match the activities of the first model to the activities of the second one. For this, we will choose an appropriate matching algorithm among the algorithms proposed in the literature. There are many such as: the greedy, the A-star, the exhaustive and the process heuristic algorithm [15]. The greedy and the A-star ones offer the most interesting tradeoffs. The A star algorithm offers a slightly better mean average precision but is significantly slower.

5. Comparison of transitions between the two models: This step consists of finding the number of transitions that exist in the equivalent model after the matching of activities. Simply we consider \(L\) a list of pairs \((A, B)\) with \(A\) and \(B\) two distinct activities in the reference model, as there is a transition from \(A\) to \(B\). Identically, in the other model we consider \(L_{\text{ref}}\) a list of pairs \((C, D)\) with \(C\) and \(D\) two distinct activities in the business process model as there is a transition from \(C\) to \(D\). We check then for each pair \((A,B)\) in \(Lm\) is there a pair \((C,D)\) in \(L\) where \(A\) is equivalent to \(C\) and \(B\) equivalent to \(D\) referring to the matching of activities previously done. The number of equivalent transitions is increasing simultaneously as the condition is verified. By having this number, we collect all the necessary information to decide if the process model refers or not to the process model with which it
has been compared. If it doesn't refer, the matrix of distance will help to determine which activities and transitions must be changed in order to improve the existent process.

4. Developed prototype

There are many areas in the industry in which business processes can be subject to a BPR project. The life cycle of computer software is a typical example that we can study as we notice a significant increase in the number of software development companies interested in standardizing and improving their work processes. From here comes the idea to apply our novel approach of comparing business processes presented in the previous section in software development sector. We called the prototype "BPMComparer". This application will compare two models of business processes in XPDL formats. The prototype is a WinForm application implemented with visual studio 2008 written in c# language. The company begins with describing its business process in BPM Notation. This task can be done by a specialist or a simple responsible using one of the various and available BPMN editors such as Aris Express, Open ModelSphere, Bonita or BizAgiProcess Modeler which we used in our experimentation. These editors offer the possibility to get the XPDL format of the model. In fact, we chose XPDL because it's an easy exploiting format and the main advantage of it is that it is a standard defined by a well-established authority (the WFMC) and which uses the industry-standard XML technology which remains intelligible without the need for a parser and modeling software [16]. The structure of the file is, as described in the Figure 2, mainly composed of activities and transitions tags. The prototype named "BPM Comparer" has an interface simple to use and does not require any expertise. The functioning of prototype is presented by the use case diagram shown in Figure 3. To enter the Model to the application, "BPM Compare" gives the possibility to choose the XPDL file in a directory on the work station, or to enter its path directly. Then the user chooses a reference model to compare with, displayed in a list box. If the path given is incorrect, the file is not well formed or even if simply not mentioned an alert is displayed to the user and the process is blocked until it is corrected. With .net libraries, we could parse the XPDL files and display the contents of activities' tags and transitions' tags without considering all the tags concerning the displaying of the elements in the BPMN model (the positions and sizes ...). From these sets of labels, a matrix of distances is established. Syntactic distances are values from 1 to 1000. The value is bigger as the difference
between the activities is. Then this matrix is updated when semantic similarities are calculated. If the two labels are synonyms, the distance is set to 1. We chose the dictionary offered by Microsoft Word as it is rich enough and has an extensive use. In addition to that, .net offers a dll to exploit Microsoft Office tools. This is one choice, and not the only available: we can use another free published thesaurus or create our own synonyms' dictionary according to the application domain.

Fig. 2 Logical XDPL structure

After that it comes the matching step for which we implemented the greedy algorithm. This algorithm starts by marking all possible pairs of nodes from the two graphs as open pairs. In each iteration, the algorithm selects an open pair that most increases the similarity induced by the mapping, and adds this pair to the mapping. The selected pair consists of two nodes. Since each node can only be mapped once, the algorithm removes from the set of open pairs, all pairs in which one of the selected nodes appears. The algorithm iterates until there is no open pair left that can increase the similarity induced by the mapping.

Fig. 3 Use case diagram of “BPM Comparer”

The limit number of equivalent transitions and the difference between the number of models' activities, which are essential indicators to decide whether the models are similar or not, are parameters that users can configure to the application.
5. Experimentation

Our approach has been tested and experimented with models collected from some IT projects team leaders in several software companies, the pioneers in Tunisia. The activities are labeled in French and we used therefore the French dictionary of Microsoft for semantic comparison. Figure 4 illustrates the first case of study; whereas Figure 5 illustrates the reference Model known as the Water-Fall Model.

Fig. 4 First business process studied

Fig. 5 WaterFall Model

The process presented in Figure 5 consists in the following steps: specifications, architectural design, detailed design, implementation, testing and finally integration. The cycle can be repeated from beginning or just from the implementation step according to the feedbacks of the customer. The main reason for adopting such a process of development is the nature of the projects: critical requirements, long-term implementation, their integration and validation could be in a specific environment with expensive costs...

The Water-Fall Model was developed since 1966 then formalized in 1970. It defines sequential steps. For each step documents are produced to verify conformity before moving to the next [17]. We convert the first model to XDPL format.

Once the comparison launched, the application begins the extraction of data(Activities and transitions) from the models and displays it as shows the screenshot in Figure 6.
The application computes the matrix of distances between labels, match the activities and calculate the number of equivalent transitions. Based on these information, it decides whether the models are similar or not and displays all the results of the comparison (Figure 7) which can be of a help for the experts. In this case labels are mostly equivalent and we can consider that the models compared are similar. Even though this comparison indicates compatibility between the two models, the table of results can help to improve the process to make it more conform to the model of reference. The same experience was made when comparing another process whose model is illustrated in figure 8, with the V Mode [19].

The importance of the prototype is touched when comparing the third model (Figure 9) with the Water-Fall Model. Actually, we cannot consider them as similar. The extracted data, calculated distances and other details' analysis will guide experts to find exactly the steps or the tasks that they have to review. Even better, this experience can push the company to think seriously about launching a BPR project.
6. Conclusion and future work

In this paper we have presented an approach for measuring similarity between business process models. We have shown that calculating only syntactic similarity is insufficient since the instance context is not considered and homonyms cannot be discovered. By taking into account the signification of activities' labels and the different transitions, sufficient similarity degrees between element names and processes can be computed. This has also been demonstrated in a case study related to software development in which we experimented a prototype based on the novel approach. This prototype compares two software development process models: an existing one to a reference one. These models are designed with BPMN [18]. The simplicity of our approach makes it possible to apply our similarity calculation for other process modeling languages too. Our similarity measurement approach provides a basis for future research. In fact, we will investigate the benefits of our approach in various case studies. Furthermore, there are some practical functionalities which can be added to the prototype and can for example give the possibility to the user to enter new models to compare with and save them. Once this functionality developed, companies will find it easier to maintain large databases of process models.

We also focused on similarity of business processes with respect to tasks and relations between tasks. Other aspects of business processes can be considered when determining similarity, e.g. used or generated resources. Besides, process models can be annotated with information that helps to determine the similarity more precisely, such as textual documentation. Exploiting such additional information is an avenue for future work.
References


