Application of Heuristics in Business Process Models to Support Software Requirements Specification

Fernando Aparecido Nogueira and Hilda Carvalho de Oliveira
Institute of Geosciences and Exact Sciences, São Paulo State University (Unesp), Rio Claro, Brazil

Keywords: Requirements Engineering, Business Process Model, BPMN, XPDL, Heuristics, UML.

Abstract: Requirements Engineering has suffered difficulties caused by communication failures between business and Information Technology teams (IT). The knowledge of the enterprise's business domain is very important for the systems analysts when developing software solutions to automate activities and processes. However, these analysts come across frequent changes in the system scope and requirements descriptions incomplete or erroneous. This work presents a systematic process that takes into account the business process models to automatically extract functional and non-functional requirements, which compose the Software Requirements Specification document. This automatic process uses requirements heuristics implemented by a freeware software system that generates software requirements documents with use cases and UML diagrams. The systematic process uses XML to facilitate systems integration, as well the re-use and visualization of the results. Additionally, this work presents business heuristics that enable significant improvements in the documentation of the business process models, bringing advantages for the business and IT. Assessment tools for the level of documentation level are proposed for both the business process models as for software requirements documents.

1 INTRODUCTION

The knowledge about the organization's business domain is very important for systems analysts when developing software solutions to automate activities and processes. Kalinowski et al. (2015) observe that the Requirements Engineering activities have suffered from the difficulties caused by communication failures between the Business and Information Technology (IT) teams. These teams use different vocabularies, languages and technical models. This hinders the elicitation and modeling software requirements (Bousetta et al., 2013). Although there are standards such as ISO/IEC/IEEE 29148:2011, Mafra et al. (2016) note that there is a lack of a standard for the software requirements specification document. The consequence of all these problems can be seen in the frequent changes in the system scope and the description of incomplete or erroneous software requirements.

This paper presents a solution that defends that software requirements can be determined more assertively if the organization's business process models are considered. The more documentation software requirements are similar to the organization's processes, the greater the level of compliance of these requirements with the customer's needs (Vieira et al., 2012). Consequently, the positive effects will affect the next stages of software development. Thus, the software product delivered to the end client will meet more effectively the needs of users and stakeholders of the organization.

Well-defined business processes provide a better understanding of what the company does in their business and how the process is executed in different departments. The cross-interaction between their different organizational divisions is documented by the business process models. These models provide a clear scenario for improvements in the processes execution, considering the use of software and other resources. This has helped to emphasize the adoption of the BPM (Business Process Management) approach, which involves the analysis, definition, execution, monitoring and management of business processes (Van der Aalst, 2013). For this, the BPMN business process models are essential. They can be built by using various languages and notations, but the graphical notation BPMN (Business Process Model Notation) is considered the standard of the current market. According to BPMN specification maintained by the

Nogueira, F. and Oliveira, H.
Application of Heuristics in Business Process Models to Support Software Requirements Specification.
DOI: 10.5220/0006280400400051
In *Proceedings of the 19th International Conference on Enterprise Information Systems (ICEIS 2017) - Volume 2*, pages 40-51
ISBN: 978-989-758-248-6

Consortium OMG (Object Management Group), textual documentation of its elements can make to support the modeling of business processes.

In this context, this paper presents a systematic process that takes into account the business process models of an enterprise to extract requirements for the systems analysts team, maintaining the vocabulary and language used in the business environment. The process automatically extracts functional and non-functional requirements from the business process models in BPMN v2.0 notation. For this reason, other important instruments have been developed to support systems analysts. Because of the need for information to compose the Software Requirements Specification document, some instruments to improve the documentation of the BPMN elements have also been developed. This way, the organization gains with a more knowledgeable IT team in the business environment, with better software to their needs and with better-documented models in business processes.

In order to apply this systematic, the business process models in BPMN v2.0 must be converted to XPDL (XML Process Definition Language) v2.2 using modeling systems available in the market. XPDL establishes a structured format based on XML (eXtensible Markup Language) for representation of process definitions. Thus, considering the specification of single metamodel, different modeling tools can interact and generate documents in the XPDL standard from BPMN models, without losing information in this transformation process (Van der Aalst, 2003).

The extraction of information from business process models in XPDL is possible with the support of heuristics set ("business heuristics") which helps to identify the elements of functional and non-functional requirements. The extracted content is structured in a requirements document in XML, containing use cases and diagrams in UML (Unified Modeling Language).

For the automation of the presented process, it was developed a software system called SRPD (Software Requirements from Process Definitions). The inputs are XPDL files that represent the business process model and the outputs are the functional and non-functional requirements document in XML, that facilitates the viewing. These documents comprise the initial version of the Software Requirements Specification, in order to ease the communication between the requirements analysts and the company and to improve the time spent in the specification of software requirements. In addition, it is worth mentioning that this paper presents a feature that helps analysts in evaluating the level of information in this document.

Based on this, Section 2 presents some works that also use business process models to support the elicitation of software requirements, although in a different and more limited way. Section 3 presents concepts related to business process models in BPMN and XPDL, which are relevant to the job. The systematic process for automatic extraction of software requirements from business process model is presented in Section 4. This section includes various tools and heuristics used in the systematic process as well the SRPD system. Section 5 presents the application of the proposed methodology, using real business process models. The final considerations and perspectives for future works are presented in Section 6.

2 RELATED WORK

There are some works in the literature that also present approaches for extraction of software requirements from business process models of an enterprise. These papers differ in many aspects, such as modeling notation, complexity, business processes details, software requirements handling, among others. However, none of them proposes the automatic extraction of requirements in the same way as this work does. Although some studies use heuristics, they differ from the heuristics defined in this paper regarding its type, purpose, and scope.

Some papers consider BPMN notation for modeling business processes, such as Xavier et al. (2010), Bousetta et al. (2013), Vieira et al. (2012), Macek and Richta (2009) and Cruz et al. 2014). However, they do not use the conversion of these business process models to XPDL to extract software requirements, as proposed in this paper. Moreover, Dias et al. (2006) explore the relationship between activities from business processes and use cases. However, business process models are designed using UML language instead of BPMN.

Bousetta et al. (2013), Dias et al. (2006) and Cruz et al. (2014) use sentences in pre-defined formats for the business process activities documentation. Therefore, it requires that the process modelers and specialists are aware of these sentence patterns instead of using their own business language. For instance, Bousetta et al. (2013) propose the use of a standard documentation based on the concepts of "term" and "fact" to describe business rules in business process models. This standard allows the use of natural language and aims to assist the generation of use cases, class and sequence diagrams using UML. It is worth mentioning that this is executed in a semiautomatic mode in which the authors use a software tool, however, it is necessary to adjust manually the diagrams generated at the end of the process.

In another direction, Xavier et al. (2010) highlight that the BPMN notation does not provide elements to represent explicitly non-functional requirements. Then, the authors propose a technique that extends the BPMN notation, placing labels on the activities that present non-functional requirements. Non-functional requirements are also considered in this work's guidelines. They are described by using "extended attributes", which are available in BPMN and must be added to the appropriate elements of the business process models, as will be presented in Section 4.

Vieira et al. (2012) use instructions to guide the systems analysts during the elicitation of requirements from the activities of business process models. These instructions are called by the authors as "heuristics" and they are performed manually. It is noted that the business process models are designed by Vieira et al. (2012) using BPMN. The use of heuristics gives support to register, validate and repeat the procedures performed. However, the application of heuristics is restricted to a very small set of requirements by Vieira et al. (2012). In addition, the heuristics do not consider resources of the business process modeling tools, such as the indication of the activity performer (actor) and the use of extended attributes. However, the proposal presented in this paper, consider these business process modeling tools features. As shown in Section 4, this work also uses heuristics to extract requirements from business process models, called "requirements heuristics". These heuristics are principles automatically identified in the corresponding XPDL code of the business model, allowing the automatic generation of a requirements document.

Following this idea, it is emphasized that only the procedures used by Dias et al. (2006) are performed automatically. The authors use a specific software tool that enables to model business processes in UML and makes the extraction of requirements by transforming them into UML diagrams. The procedures are performed in a semi-automatic way by Bousetta et al. (2013) and manually in the other works mentioned in this section, impacting the time required and the results obtained.

The works of Bousetta et al. (2013), Dias et al. (2006) and Cruz et al. (2014) generate UML diagrams for use cases, while Macek and Richta (2009) only generate activity diagrams. These papers do not use any structure for the Software Requirements Specification (SRS) document. This would help the business and IT teams to understand and validate the software requirements. On the other hand, the purpose of this work generates a software requirements document according to the ISO/IEC/IEEE 29148:2011 standard and best practices of Requirements Engineering. The requirements document is generated in XML, using a metamodel, which defines a hierarchical structure for the identified requirements representation.

In a different direction, Herden, Farias, and Albuquerque (2014) present a way to support the prototyping phase in the agile approaches with use case descriptions using the BPMN notation. However, the authors do not explain how the information handled by the process activities are represented. On the other hand, the proposal presented in this paper uses the following BPMN elements for representing such information: artifacts and data repositories. It is appropriate to mention that Inayat et al. (2015) point out that the use of agile methodologies can present negligence about the non-functional requirements.

3 BUSINESS PROCESS MODELS

According to the CBOK (Business Process Management Common Body of Knowledge) guide (ABPMP, 2013), business process modeling is the set of activities involved in creating representations of the existing or proposed business processes in a complete and accurate way. These activities are critical to the management of an enterprise, supporting the understanding, communication, and management of the business processes (ABPMP, 2013). The representations can be done by using graphical notations, although additional textual information is important to document the resulting models.

According to Jung et al. (2004), business process modeling can be done in two different ways: using a graphical notation such as BPMN, EKD, and UML, or through an execution language, such as BPML (Business Process Modeling Language), BPEL (Business Process Execution Language) and XPDL. Generally, these languages are based on XML and allow the understanding of information that make the process run. Considering the purposes of this paper, subsections 3.1 and 3.2 approaches BPMN and XPDL, respectively.

3.1 Considerations on BPMN

The OMG Consortium is responsible for the BPMN notation specifications. This paper considers the latest version, v2.0, published in 2011, which includes seventy-five graphic elements. It is a comprehensive and flexible notation that can be used by professionals with different levels of expertise. According to OMG (2011), BPMN is currently the business process modeling notation used by most of BPM professionals.

The full specification of BPMN v2.0 defines attributes grouped into four basic categories of elements: objects, connecting objects, swimlanes and artifacts. Flow objects are the graphic elements represent events, activities, and gateways (for decisions). Connecting objects interconnect Flow Objects through different types of arrows. Swimlanes (Lanes and Pools) group activities into separate categories, according to their functional capabilities or responsibilities. Artifacts add information to the process, such as processed data or comments. Fig. 1 illustrates a business process diagram, with some elements of BPMN notation identified through information notes, highlighted in red.

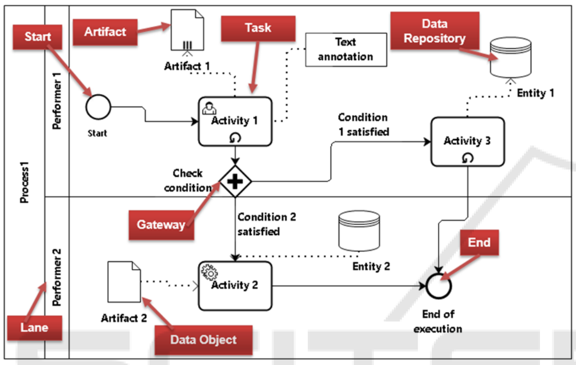


Figure 1: An example of a business process diagram in BPMN.

It is worth mentioning that OMG does not specify the rules nor the elements for the documentation of business processes. Because of this gap, business process modelers systems in BPMN, which are part of complete BPM systems, must implement effective ways to document the elements of the business process models. This documentation is critical to the BPM lifecycle. However, the diversity of modeling systems brings problems when exporting the complete model (graphics and documentation) to other modeling systems.

Another issue that deserves attention is the lack of a guide of procedures for business process modeling using BPMN. However, some studies suggest good modeling practices so that the integrity and logic of the processes execution are maintained (Correia and Abreu, 2015).

3.2 Mapping from BPMN to XPDL

The Consortium Workflow Management Coalition (WfMC) is responsible for the XPDL language specifications. This paper considers its latest version, v2.2, published in 2012, the business process defined by the XPDL structure can be interpreted by different software tools. According to the XPDL specification, the BPMN business process modeling tools should provide two operations: (1) export a business process model to XPDL, according to the tool's internal representation for the business process definition; (2) importing a business process definition from the corresponding XPDL code. To perform these operations, the BPMN modeling tool should conform to the XSD (XML Schema Definition) schema provided by the XPDL specification.

The XPDL language specification defines a basic set of entities and attributes for representing lanes, pools, processes, participants and message flows, as well as different types of elements that represent the flow control in the business process models. The specification also defines the serialization of the business process models in XML format, including conceptual and graphical information for the representation of the processes (Weske, 2012).

Fig. 2 shows an example of the structure of a business process in XPDL. It is possible to identify the workflow process, which is the process definition for the "Process A" process. Then, the activities identified as "A" and "B" are represented, highlighting the "Transition" element (identified as "AB"), which connects them.

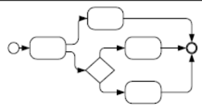
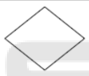
```
<WorkflowProcess Id="Process A">
  <Activities>
    <Activity Id="A">
      ...
    </Activity>
    <Activity Id="B">
      ...
    </Activity>
  </Activities>
  <Transitions>
    <Transition Id="AB" From="A" To="B"/>
  </Transitions>
</WorkflowProcess>
```

Figure 2: An example of a business process in XPDL.

For this paper, business process models in BPMN are exported to the XPDL format. Afterward, these models are validated and processed by a software tool presented in Section 4, which is responsible for the automation of the proposed systematic process.

The OMG and WfMC consortia do not define rules for mapping the elements of BPMN and XPDL models. The modeling tools must define and implement how to associate each BPMN element with an XPDL element (and vice versa). Due to the conceptual difference between these models, some graphics are not used in XPDL, although the modeling tools export them. Table 1 shows the main elements used in this study to relate BPMN and XPDL elements, based on the works of Van der Aalst (2003), Mora et al. (2007) and White (2003). The complete set of mappings used in this work and the description of requirements heuristics will be presented in Section 4. This mapping does not include all associations between BPMN and XPDL elements but is sufficient for the extraction of requirements proposed in this paper.

Table 1: The mapping between BPMN and XPDL elements (White, 2003).

BPMN Graphical Object	***Mapping to XPDL***
 The details of a Pool or an Expanded Sub-Process	<WorkflowProcess/>
Start Event	<Activity> <Route/> </Activity>
	<Transition/>
Task	<Activity> <Implementation> <Tool/> <Performer/> </Implementation> </Activities>
 Decision	<Activity> <Route/> <TransitionRestriction> <Split Type="XOR"/> </TransitionRestriction> </Activities> Combined with a: <Transition> <Condition/> <Transition>
End Event	<Activity> <Route/> </Activity>

4 PROCESS FOR EXTRACTION AND SPECIFICATION OF SOFTWARE REQUIREMENTS

This section presents a systematization of the actions to extract functional and non-functional requirements from business process models of an enterprise, to support the development of software solutions for process automation. Due to the complexity of a business process model, with several sub-processes, it is recommended that the application of the systematic process is done separately for each sub-process. The application of the systematic generates a requirements document, which is structured according to the guidelines in the ISO/IEC/IEEE 29148:2011. Eventually, the requirements documents generated in each application of the systematic process must be analyzed to reach a set of requirements that is in accordance with the scope of the software to be built. The final document is an early version of the Software Requirements Specification, which will support systems analysts to understand the business.

In order to apply the systematic process, the company must have well-defined processes designed in BPMN notation v2.0, including descriptive and textual parts of the model elements. The model must have a start event, at least one end event, at least one lane and one pool. The textual part of the model should provide sufficient information to understand the implementation particularities of the company's business processes. This information must be clear and accurate, covering the roles of the processes performers, the documentation of the BPMN elements in the chain of processes, existing software systems, and other information relevant to the processes.

However, due the lack of a standard for the textual description of the model and often the lack of skills and time of the business process modelers, a considerable part of all these information is missing or incomplete in the business models. So, before applying the systematic process, it is important that these amount of information is properly present in the business process models.

In this work, the extended attributes feature of BPMN notation was used to facilitate the identification of some software elements. These attributes are structured according to the "key-value" pattern and can be added to the documentation of any element of a BPMN model using the modeling system interface. Three types of extended attributes were defined in this work, with the following identifications: "NFR", to be added to the activities or decision flows, representing non-functional requirements; "ATTRIBUTES", for data repositories or artifacts, representing their attributes and respective data types; "RULE", for the business process activities, representing the related business rules.

The proposed systematic process is organized in three steps that must be performed sequentially in this order:

- STEP 1: exportation of the BPMN v2.0 model to XPDL v2.2 using a modeling tool for business processes. At this step, the input is the business model in BPMN, and the output is composed of XPDL files, where an XPDL file is generated for the whole model and a file is generated for each sub-process model if any;

- STEP 2: execution of the SRPD system for each XPDL file created in Step 1. The output is a requirement document in ".xml" file extension, which shall form the Software Requirements

Specification (SRS) document. An XSLT (eXtensible Stylesheet Language for Transformation) stylesheet is attached to the ".xml" file for formatting the requirement document and thus improve the presentation of the content of this document. The SRPD system (Fig. 3) performs the following activities in this order:

2.1 - automatic extraction of functional and non-functional requirements using a set of heuristics ("requirements heuristics"), presented in Subsection 4.1;

2.2 - automatic generation of an ".xml" file with the specification of the use cases identified and the following UML diagrams: use cases, classes, and activities. The structure of the document and the generation of diagrams are discussed in Subsection 4.2.

- STEP 3: composition of the SRS document. The inputs are the requirement document (".xml") generated in Step 2. The systems analysts team, to enable further adjustments, must compose this initial version of the SRS document.

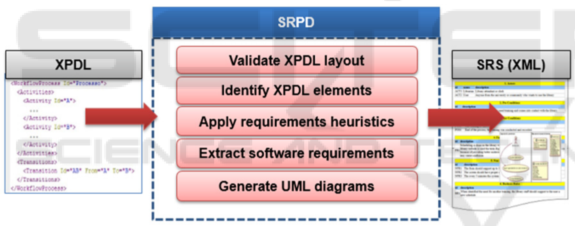


Figure 3. Overview of SRPD software.

It is recommended to repeat the implementation of the systematic process if any changes are made in the business process models. However, if the change is made in a sub-process, the systematic process shall be applied again only in the XPDL file for the sub-process. Thus, the requirements documentation will be updated and aligned with the company's business processes. It is worth mentioning that Step 1 is the only step in which manual intervention is required. Step 3 can be performed automatically, although this is not discussed in this article.

4.1 Automatic Extraction of Software Requirements

The input for the extraction phase for software requirements is a XPDL file that represents a business model for a process or sub-process. The SRPD system inspects the XPDL file and applies nine heuristics, called "requirement heuristics." These heuristics enable a mapping for the textual elements that will compose the requirements document (part of the Software Requirements Specification). Table 2 shows the correspondence between the BPMN elements used in this work, XPDL structures and the software requirements elements generated. It is important to observe that all the associations between BPMN and XPDL elements presented in Table 2 have been considered in the automatic conversion performed by the modeling tools analyzed.

Table 2: The relation between BPMN, XPDL and software elements.

BPMN element	XPDL element	Software element
Diagram	WorkflowProcess	Scenario
Performer	Performer	Actor
Activity	Activity	Functional requirement / use case / activity
Decision Flow	Route	Business rule
Artifact	DataObject	Class and attributes
Data Repository	DataStoreReference	Class and attributes

The requirement heuristics are classified according to the software requirements elements. The three categories of these heuristics are shown in Tables 3 to 5, respectively: heuristics for extraction of use cases scenario, heuristics to extract textual requirements and heuristics to generate UML diagrams. Requirements heuristics are identified by "RH" and a sequential number. Each heuristic presents a principle that is considered in the SRPD system instructions.

Table 3 shows that the RH1 heuristics are used to identify the event that starts the process, to obtain a pre-condition for the use cases. The documented information about this event will be mapped to the "Pre-condition" section of the document. Similarly, the RH2 heuristic is used to extract the post-conditions from the documentation of the process end events.

The application of RH4 heuristic depends on the annotation "NFR" in the documentation of the business process models. It is observed that the inclusion of multiple non-functional requirements in the same activity, must use the character semicolon (";") to divide the non-functional requirements.

The RH5 heuristic uses the documentation of the decision flows of the business process diagrams to extract the software business rules. Decision flows represent decision-making to perform activities of one or more different paths. The constraints associated with decision-making must be documented in the respective decision flows of the model using natural language, according to the organization's business knowledge.

Table 3: Requirement heuristics to extract use cases scenarios.

Heuristic	Description
RH1 pre-conditions	– The start event element must be identified. – The precondition for performing the business process must be registered in the event documentation.
RH2 Post-conditions	– End events must be identified. – The documentation of the end event must represent the post-condition required after the execution of the business process. – The post-condition must be registered in the “Post Conditions” section of the SRS document.

Table 4: Requirements heuristics to extract textual requirements.

Heuristic	Description
RH4 Non-Functional Requirements	– The type of the activity must be identified: “User”, “Service” or “Script”. – The extended attributes of the type “NFR” must be identified in the activities.
RH5 Business Rules (decision flows)	– The decision flows must be identified. – The business rules must be identified from the documentation of the decision flows.
RH6 Business Rules (activities)	– The activities with the types “User”, “Service” or “Script” must be identified. – The business rules must be identified from the extended attributes of the type “RULE”.
RH8 Functional Requirements	– The type of the activity must be identified: “User”, “Service” or “Script”. – The functional requirements must be identified from the textual documentation of the activities.

On the other hand, there are business rules that are not documented in the decision flow of the business processes, for example, when a given task must be performed in a certain time by a given user. This restriction should be recorded in their activity with the insertion of extended attributes of the type “RULE”. Then the RH6 heuristic can identify these attributes in the business process activities.

The RH3 heuristic aims to establish a relationship between the role of the activities in the business process and the use cases. Similarly, the RH3 heuristic aims to relate the role of the business process activities and functional requirements of the software. For this, RH3 and RH8 identify the activities of the business processes that are "User", "Service" and "Script". The activities of the type "User" represent the execution of a task by a human being with the help of a computational tool. "Service" represents actions performed by a computer system only. "Script" are performed using mechanisms created in a language that is understood by the business process.

Table 5: Requirements heuristics to generate UML diagrams.

Heuristic	Description
RH3 Use Cases	– Use cases must be identified from activities with the type “User”, “Service” and “Script”. – Actors must be identified from the resources that perform the activity.
RH7 Domain Classes	– Domain classes must be identified from each artifact and data repository. – Class attributes must be identified from the extended attributes of the type “ATTRIBUTES”.
RH8 Activities Diagram	– The activity diagram start must be identified from the start event of the business process. – The activities of the activity diagram must be identified from the activities with the types “User”, “Service” and “Script”. – The end of the activity diagram must identify from the end event of the business process.

Thus, functional requirements are identified and recorded in the requirements document, at "Functional Requirements" section, considering the documentation of each activity. The generation of the use case diagram in UML is started when the executors of each activity are identified; they will be the actors in the use case diagram. This identification is provided in the BPMN notation and XPDL language by using information the "performer" information". The identified actors are recorded in the "Actors" section of the document. The illustrations of the use cases are generated from the textual description of each activity selected and are represented with the respective actor who implements it.

The identification of the software domain classes is performed by RH7 heuristic (Table 5), using the structured documentation of data repositories and data objects from the business process diagrams as well as the associations of these elements with the activities. These elements represent information storage objects that are managed during the execution of the business process. Each data object and each data repository correspond to a domain class. The attributes representation of the identified classes is possible with the use of the “ATTRIBUTES” extended attributes in the documentation of these elements in the business process models. These extended attributes must record the data attributes types of each data object or data repository so that the area of representation is possible by RH7 heuristic.

The definition of a formal sentence for representing the attribute name and its respective data type meant to reduce problems arising from the use the natural language in the identification of these attributes process. In addition, a dictionary of terms for the representation of data types was defined. This dictionary was created for two reasons: ease the registration of the data types during the business process modeling and restrict the use of natural language. The data types of the attributes from the class diagrams present in the software requirements document are primitive type commonly used in programming languages, as shown in Table 6. The information about the identifier and data type of each attribute is found in the documentation of the extended attribute of the type "ATTRIBUTES" in the model in BPMN. For example, if the documentation of the extended attribute "ATTRIBUTES" has the text "status: B;" then the class attribute is identified as "status" with the data type "boolean".

Table 6: The domain of data types for the design of UML activity diagrams.

Symbol	Description	Datatype in class diagram
C	Char	char
T	Text	string
D	Decimal	float
I	Integer	int
B	Boolean	boolean

The data types of the attributes from the class diagrams present in the software requirements document are primitive types commonly used in programming languages, as shown in Table 6. The information for these attributes is in the documentation of the extended the attribute with the annotation "ATTRIBUTES "in the BPMN model. The identification and type of each attribute of the class diagrams depend on the documentation of the extended attribute "ATTRIBUTES". For example, if the documentation of the extended attribute "ATTRIBUTES" has the text "status: B;" then the class attribute is identified as "status" with the data type "boolean".

4.2 Software Requirements Document

The requirements document generated by SRPD system is structured according to the recommendations of the ISO/IEC/IEEE 29148:2011 standard. This document ".xml" also includes contents indicated by Pressman (2015), Sommerville (2015) and Yayici (2013). The requirements document consists in the following sections, with the structure shown in Fig. 4:

- **Versioning:** information about the considered business process models;
- **References**: provides information about the XPDL document that contains the definition of the business process;
- **Software features**: textually describes the actors, functional requirements, non-functional requirements, and business rules; it also presents the pre-condition and post-conditions for implementation of use cases;
- **Requirements model:** presents the following UML diagrams: use cases, classes and activities.

Regarding the traceability of software requirements, a relationship between business process models and the extracted the software requirements has been established. Textually, the features traceability, sources, and dependencies are recorded is registered in the generated requirements document.

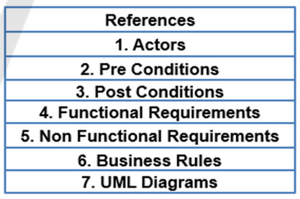


Figure 4: The structure of the requirements document.

Each requirements document is associated with an XPDL file, which represents a subprocess or the entire business process model in BPMN. Thus, more than one requirement document can be generated for a business process model. These documents must be gathered and organized by the systems analysts team to compose the Software Requirements Specification (SRS) document. Other documents and information can complement this document, which will form the basis for the alignment of the communication between business and software development teams.

Fig. 5 shows a metamodel in UML for the definition of the requirements classes considered in the requirements document. This metamodel allowed to create an XSD schema for serialization of the document in XML format - which enables its use in other phases of the software development (e.g.: design and testing).

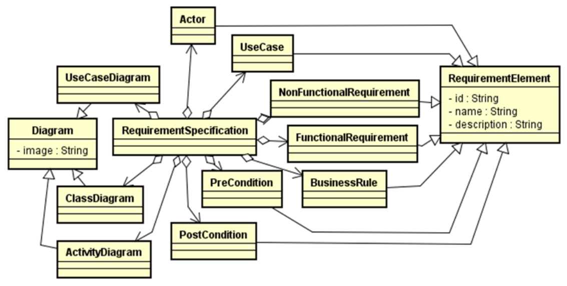


Figure 5: Metamodel for the requirements document.

It is worth mentioning that the activities of the business process models processes can indicate the use of software systems that are already in operation in the process chain. In these cases, the requirements document generated contains classes that represent information that is maintained by these systems and used in the implementation of the company's business processes. Moreover, the use cases identified may help to identify the activities that are already automated by these systems. This supports the process of requirements analysis, identifying what should be automated by a software solution to be developed. It is also possible to identify some characteristics of the interaction between new systems and the already operational ones, aimed at automation of the business processes.

4.3 SRPD System for Automation of Software Requirements Extraction

The SRPD (Software Requirements from Process Definitions) system was developed for automation of Step 2 in the proposed systematic process. The input is an XPDL v2.2 file corresponding to the business process model or a subprocess of this model in BPMN v2.0. The output is a requirements document according to the structure presented in the previous subsection. The SRPD system can be requested free of charge for use and evaluation. The system was developed in Java SE (Standard Edition) V1.8 and can be executed on Windows, Linux and Mac platforms.

The SRPD consists of two modules (Fig. 6):

- **XPDL Parser**, which parses the XPDL v2.2 code and uses the requirements heuristics, which were defined in Subsection 4.1;
- **XML Generator**, which generates requirements document in XML and one XSLT style sheet, in order to provide the visualization of the XML document and the UML diagrams using Web browsers.

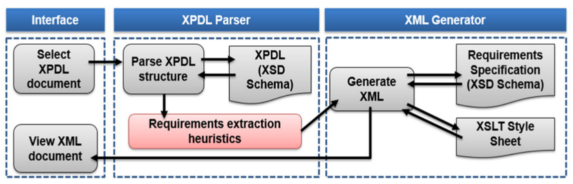


Figure 6: Architecture of the SRPD tool.

The library PlantUML (Plantuml, 2012) was used to generate the UML diagrams It is a free and open source library that uses a proprietary language to generate several UML diagrams. The library can be used together with many programming languages and development tools. In this work, the diagrams were generated in PNG (Portable Network Graphics) format, suitable for the structure of the XML document generated. The user interface allows the user to select the desired XPDL file.

The progress of the process can be followed by the user through a panel, which also displays error messages in case of failure.

5 EVALUATION

The systematic process and the SRPD system were evaluated with business process models in BPMN v2.0 available at the public repository: http://www.bizagi.com/en/community/processxchange. In that repository, there are models with different complexities and directed to different business areas. The textual documentation of these models contributes to the understanding of the business domain of the companies for which were designed. It must be observed that the models from the repository use the modeling tool Bizagi Modeler, which offers numerous features for textual and graphical documentation models in BPMN.

The models used to evaluate the proposed systematic process have the following characteristics: - BPMN v2.0 notation; - More than one pool, which means that the process activities are performed by different departments of the organization; - Activities using actual forms and information that can be stored using databases; - Documentation that identifies business rules.

In this section, two business process models will be discussed to show the applicability of the proposal:

(1) model for the accounts payable department of a company, which has processes to validate the documentation submitted by suppliers and subsequent release of payment; (2) model for software permissions control area department in an organization.

The original models were exported to XPDL v2.2 using two freeware modelers: Bizagi and Camunda. So, each XPDL document was submitted to SRPD tool for the generation of requirements documents. The results were similar, regardless of the modeler system, without interfering in the requirements resulting documents.

The evaluation of each requirements document was performed manually, with the support of a classification presented in Table 7. This rating scale is from 1 to 4, where 4 indicates the highest level of desirable information in an element of the requirements document structure. This classification allows analyzing the adequacy of the documentation of each requirement element resultant from the SRPD system. In all cases, the results showed that most of the elements in the requirements structure have not reached the level 4, then BPMN models were not properly documented. Thus, some adjustments were necessary for these two business models, considering the insertion of elements such as data repositories, artifacts, and extended attributes. In addition, the textual documentation of the BPMN elements and the identification of the executors and types of each activity were adjusted when necessary.

The adjusted models were again subjected to the same initial procedures, showing significant improvement over the completeness and content of the requirements document. This process was repeated until the set of identified requirements were rated at level 4, that is, the documentation is considered "complete".

In addition to the models available in the repository mentioned, other models were used, highlighting the business process models of the Library of a public University, with high organizational complexity. The models of this Library used a variety of BPMN v2.0 notation elements which are not normally used in most of BPMN models (based on literature and public repositories). Furthermore, the models are composed of several sub-processes and include activities automated by software systems and semi-automated activities. These business process models were developed by members of a research group of the University.

The data shown in this section are related to the business process models conducting training for students and other users of the Library (Pucci, 2016). The textual documentation of the BPMN elements allowed applying the classification presented in Table 7. Most of the elements were classified as level 2 or 3, although some elements showed level 4. It must be observed that few elements were without any documentation, but most of them were not in accordance with the "term-fact" standard.

Table 7: Classification of the SRS elements.

Level	Description
1	– **Undocumented**: The requirement element was not identified using the proposed technique.
2	– **Labeled**: Only the name of the requirement element was identified. There is no detailed information in the documentation of the requirement element, or the information available is syntactically invalid, according to the “term-fact” expression; – Example: Actor 1: manager
3	– **Syntactically documented**: In addition to the name (level 1), the requirement documentation is enhanced with syntactically valid textual information, according to the use of the “term-fact” expression. – Example: Actor 1: Financial Manager → The financial manager is in charge of the Financial and Accounting departments.
4	– **Complete**: In addition to the level 3 premises, the requirement element must have information for the questions of the 5W1H technique, considered in the business heuristics. – Example: Actor 1: Financial Manager → The Financial Manager is in charge of the management of the activities of the Financial and Accounting departments. The Financial Manager is hierarchically below the President of the company. Everyone that works at these departments is subordinated to Financial Manager.

The original BPMN model was exported to XPDL. As it has four sub-processes, five XPDL files were generated (one for the main model). All underwent SRPD system for document generation requirements. The requirements were classified according to Table 7 and it was recorded for later comparison.

As an example in this section, the sub-process on individual training or couple will be considered. In this case, no requirement reached level 4, demanding adjustments in the BPMN model documentation. This involved the analysis of the documentation of the BPMN elements together with the staff that design the business process models. Thus, adjustments were made in graphic and textual form. An example of adjustment was to represent the forms to request training as data repositories, with attributes to register the information considered in the process: validation, scheduling, and execution.

After the application of the adjustments, the adjusted model (Fig. 7) was converted to XPDL and then submitted to SRPD tool. Figs. 8 and 9 show respectively an overview of the requirements document and UML diagrams generated. The requirements were classified according to Table 7 and most reached level 4, "complete". This was recorded for comparison with previous results.

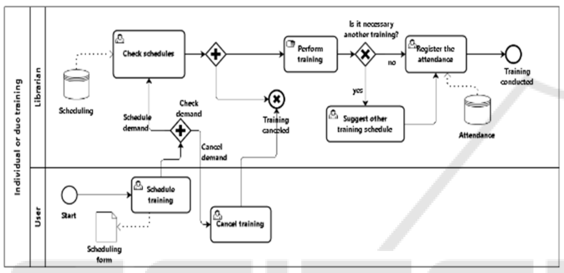
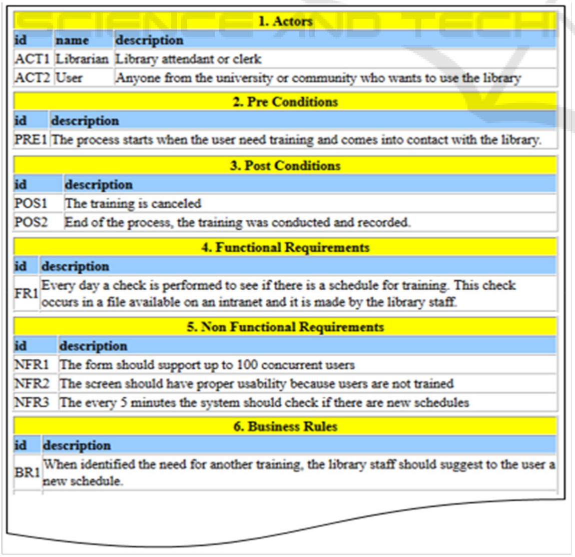


Figure 7: Overview of the BPMN model for the Library training schedule.



1. Actors		
id	name	description
ACT1	Librarian	Library attendant or clerk
ACT2	User	Anyone from the university or community who wants to use the library

2. Pre Conditions	
id	description
PRE1	The process starts when the user need training and comes into contact with the library.

3. Post Conditions	
id	description
POS1	The training is canceled
POS2	End of the process, the training was conducted and recorded.

4. Functional Requirements	
id	description
FR1	Every day a check is performed to see if there is a schedule for training. This check occurs in a file available on an intranet and it is made by the library staff.

5. Non Functional Requirements	
id	description
NFR1	The form should support up to 100 concurrent users
NFR2	The screen should have proper usability because users are not trained
NFR3	The every 5 minutes the system should check if there are new schedules

6. Business Rules	
id	description
BR1	When identified the need for another training, the library staff should suggest to the user a new schedule.

Figure 8: Parts of the SRS document generated from the business process for the Library training schedule process.

Table 8 shows significant improvement in the number of requirements obtained in the adjusted model. However, the most important are that obtained a number of conditions where most 4 has reached the level "complete". Furthermore, the BPMN model had significant improvements in its documentation, contributing to the Library business processes.

In general, the results showed a considerable increase in the quantity of identified requirements after the application of adjustments to the BPMN models. Given the identification of functional requirements, it was found that the indication of the type of activity plays an important role in the improvement of processes allowing the identification of activities that will be transformed into use cases.

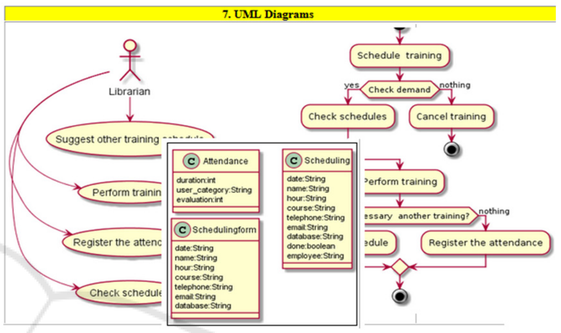


Figure 9: Detail of the UML diagrams generated from the business process for the library training schedule process.

Table 8: Summary of the results for the Library training model.

Model version	Func. Req.	Actor	Non Func. Req.	Bus. Rules	Use Case	Dom. Class	Dom. Class Attr.
Original	6	2	0	1	6	0	0
Adjusted	6	2	3	4	6	3	19

6 CONCLUSIONS

This paper presented a systematic process that takes into account the business process models of a company to extract requirements for the systems analysts team, maintaining the vocabulary and language used in business environment. The business process model can be composed of several sub-processes and include automated activities for an operating software in the organization. The systematic process automatically extracts functional and non-functional requirements business process model in BPMN v2.0.

For this, a system (SRPD) was developed to automatically generate software requirements documents with use cases structures and UML diagrams (use cases, classes, and activities). Efforts are being invested to enable the inclusion of other UML diagrams. The system uses a set of heuristics to inspect XPDL files that represent models of business process models. A technique that evaluates the degree of information of the requirements extracted was also defined. If the software requirements document obtained a low classification, the business analyst can complete/adjust the model, bringing benefits to the organization and improving communication with the development team.

For future works, DMN (Decision Model and Notation) and CMMN (Case Management Model and Notation) notations should be considered in business process modeling.

REFERENCES

ABPMP (Association of Business Process Management Professionals) (2013), "CBOK - Guide to the Business Process Management Common Body of Knowledge Version 3.0", available at: www.abpmp.org.

Bousetta, B., El Beggar O. and Gadi, T. (2013), "A methodology for CIM modeling and its transformation to PIM", *Journal of Information Engineering and Applications*, Vol. 3, pp. 1-21.

Correia A. and Abreu, F. B. (2015), "Enhancing the correctness of BPMN models", in Varajão, J. E. Cruz-Cunha, M. M. and Martinho, R. (Eds.), *Improving organizational effectiveness with Enterprise Information Systems*, Hershey: IGI Global, pp. 241-261.

Cruz, E. F., Machado, R. J. and Santos, M. Y. (2014), "From business process models to use case models: a systematic approach", *Proceedings of the 4th Enterprise Engineering Working Conference*, Funchal, pp. 167-181.

Dias, F., Morgado, G., Oscar, P., Silveira, D., Alencar, A., Lima, P. and Schmitz, E. (2006), "An approach for automatic transformation from business model to requirements model", *Proceedings of the 6th Workshop em Engenharia de Requisitos*, Rio de Janeiro, pp. 51-60.

Herden, A., Farias, P. P. M. and Albuquerque, A. B. (2014), "An approach based on BPMN to detail use cases", *New Trends in Networking, Computing, E-learning, Systems Sciences, and Engineering*. Springer International Publishing, pp. 537-544.

Inayat, I., Salim, S. S., Marczak, S., Daneva and M. Shamshirband, S. (2015), "A systematic literature review on agile requirements engineering practices and challenges", *Computers in human behavior*, Vol. 51, pp. 915-929.

Jung, M., Kim, H. S., Jo, M. H., Tak, K. H., Cha, H. S. and Son, J. H. (2004), "Mapping from BPMN-formed business processes to XPDL business processes", *Proceedings of the 4th International Conference on Electronic Business*, Beijing, pp. 422-427.

Kalinowski, M., Spinola, R. O., Conte, T., Prikladnicki, R., Fernandez, D. M. and Wagner, S. (2015), "Towards Building Knowledge on Causes of Critical Requirements Engineering Problems", *Proceedings of the. 27th Internacional Conference on Software Engineering and Knowledge Engineering*, Pittsburgh.

Macek, O. and Richta, K. (2009), "The BPMN to UML activity diagram transformation using XSLT", *Proceedings of the 9th International Workshop on Databases, Texts, Specifications and Objects*, Spindleruv Mlyn, pp. 119-129.

Mafra, P., Kallinowski, M., Fernandez, D. M., Felderer, M. and Wagner, S. (2016), "Towards Guidelines for Preventing Critical Requirements Engineering Problems", *Proceedings of the. 42th Euromicro Conference on Software Engineering and Advanced Applications,* Limassol, 2016

Mora, B., Ruiz, F., Garcia, F. and Piattinin, M. (2007), "Experiments on business process transformation from BPMN to XPDL" *Proceedings of the. 10th Congreso Iberoamericano en Software Engineering*,Isla de Marguerita, pp. 165-178.

PlantUML (2012), "PlantUML Language Reference Guide v.8048", available at: http://plantuml.com.

Pressman, R. S. (2015), *Software Engineering: a practitioner's approach*, 8th ed., McGraw-Hill.

Pucci, M. A. F. S. (2016), "*Business process management aimed at software engineering, with emphasis on process modeling"*.

Sommerville, I. (2015), *Software Engineering*, 10th ed. Pearson Prentice Hall.

Van der Aalst, W. M. P. (2003), "Patterns and XPDL: a critical evaluation of the XML Process Definition Language*"*, available at: www.bpmcenter.org.

Van der Aalst, W. M. P. (2013), "Business Process Management: a comprehensive survey", *ISRN Software Engineering*.

Vieira, S. R. C., Conte, T., Nascimento, R. and Viana, D. (2012), "Evaluating a technique for requirements extraction from Business Process Diagrams through empirical studies", *Proceedings of the 38th Conferencia Latinoamericana en Informatica*, Medelin, pp. 245-254.

Weske, M. (2012), *Business process management: concepts, languages, architectures*, Springer, Heidelberg.

White, S. A. (2003), "XPDL and BPMN", in Fischer, L. (Ed.), *Workflow Handbook*, Future Strategies, Lighthouse Point, FL.

Xavier, L., Alencar, F., Castro and J. and Pimentel, J. (2010), "Integration of non-functional requirements and business processes: integrating BPMN and NFR", *Proceedings of the 10th Workshop Engenharia de Requisitos*, Cuenca, pp. 29-50.

Yayici, E. (2013), *Business analyst's mentor book: with best practice business analysis techniques and software requirements management tips*, Emrah Yayici, Istambul.