UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

COMPLEXITY REDUCTION OF DECLARATIVE PROCESS MODELS USING A SEMANTIC ABSTRACTION CRITERION

Pedro Henrique Piccoli Richetti

Orientadora
Fernanda Araujo Baião Amorim
Co-orientadora
Flávia Maria Santoro

Rio de Janeiro, RJ – Brasil
Abril de 2015

COMPLEXITY REDUCTION OF DECLARATIVE PROCESS MODELS USING A SEMANTIC ABSTRACTION CRITERION

Pedro Henrique Piccoli Richetti

DISSERTAÇÃO APRESENTADA COMO REQUISITO PARCIAL PARA OBTENÇÃO DO TÍTULO DE MESTRE PELO PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA DA UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO (UNIRIO). APROVADA PELA COMISSÃO EXAMINADORA ABAIXO ASSINADA.

Aprovada por:

Prof. Fernanda Araujo Baião Amorim, D.Sc. – UNIRIO

Prof. Flávia Maria Santoro, D.Sc – UNIRIO

Prof. Kate Cerqueira Revoredo, D.Sc. – UNIRIO

Prof. Ricardo Massa Ferreira Lima, D.Sc – UFPE

Rio de Janeiro, RJ – Brasil
Abril de 201

Richetti, Pedro Henrique Piccoli.
R529 Complexity reduction declarative process models using a semantic abstration criterion / Pedro Henrique Piccoli Richetti, 2015.
123 f. ; 30 cm

Orientadora: Fernanda Araujo Baião Amorim.
Coorientadora: Flávia Maria Santoro.
Dissertação (Mestrado em Informática) - Universidade Federal do Estado do Rio de Janeiro, Rio de Janeiro, 2015.

1. Engenharia de software. 2. Processamento de linguagem natural

Dedicado à minha família.

Agradecimentos

Esta dissertação marca o fim de uma etapa muito importante em minha jornada. A partir de agora outros caminhos se abrirão em direção a novos desafios e conquistas. O autor é apenas um, mas este trabalho não é mérito de uma só pessoa. Gostaria de agradecer à minha esposa Angélica pelo incentivo e apoio durante todo o processo, aos meus pais Lourdes e Acélio (in memoriam) por terem me proporcionado condições para chegar até aqui, à minha irmã Graziela pelos sábios conselhos a respeito da vida acadêmica e de pesquisa. Devo agradecimentos também aos meus amigos, com quem sei que posso contar a qualquer hora. Agradecimentos especiais ao Guilherme e ao André pelas discussões e apoio na avaliação dos resultados da pesquisa.

A experiência adquirida ao longo do curso é de valor inestimável e vai além desta pesquisa. Pude viajar pela primeira vez para o exterior para apresentar um trabalho em um importante evento internacional. Evento esse que abriu portas para a realização de um trabalho conjunto com pesquisadores estrangeiros, dos quais gostaria de mencionar o Fabian Pittke, cujo projeto que realizamos de maneira colaborativa e as conversas sobre linguagem natural e modelos de processo de negócio permitiram enriquecer e aumentar a qualidade do meu trabalho.

Por fim, sem um norte dificilmente eu teria chegado até aqui. Gostaria de agradecer à dedicação das minhas orientadoras Fernanda e Flávia, que todo o tempo estiveram por perto indicando os melhores caminhos a seguir.

RICHETTI, Pedro Henrique Piccoli. **COMPLEXITY REDUCTION OF DECLARATIVE PROCESS MODELS USING A SEMANTIC ABSTRACTION CRITERION**. UNIRIO, 2015. 123 páginas. Dissertação de Mestrado. Departamento de Informática Aplicada, UNIRIO.

ABSTRACT

A declarative approach can be employed to describe only essential process characteristics, thus requiring explicit definition of constraints that limits process execution possibilities. This perspective is appropriated when dealing with unstructured or flexible processes. However, declarative process mining may result in complex models due the discovery of a high quantity of constraints, producing a cluttered model even for models with few activities. Excessive complexity is one of the major barriers to end users understanding software engineering diagrams and, analogously, business process models may suffer from the same problem. As abstractions are seen as an effective approach to represent readable models, showing aggregated activities and hiding irrelevant details, this work proposes to create language-independent hierarchical declarative maps using a linguistic hierarchy of activities. The proposed approach applies Natural Language Processing techniques for the construction of more abstract declarative models produced by process mining, where hypernymy and holonymy sense relations are applied for finding semantic hierarchies among words present in activity labels. The presented method was evaluated in a case study with real life data and support from domain experts. The findings showed that it is possible to generate meaningful groups by looking for the semantics of activity labels in order to create abstract process views with reduced complexity, starting from a low-level declarative map.

Keywords: Process Mining, Declarative Modeling, Semantic Abstraction.

RESUMO

Uma abordagem declarativa pode ser empregada para descrever somente as características essenciais de um processo, assim requerendo a definição explícita das restrições que limitam as possibilidades de execução de um processo. Esta perspectiva é apropriada quando se lida com processos flexíveis ou não estruturados. No entanto, a mineração de modelos declarativos de processos pode resultar em modelos complexos devido à descoberta de uma grande quantidade de restrições, produzindo modelos difíceis de interpretar, mesmo com poucas atividades. A complexidade excessiva é uma das principais barreiras dos usuários finais na interpretação de diagramas de engenharia de software, e, analogamente, modelos de processos de negócio podem apresentar o mesmo problema. Como abstrações são vistas como uma abordagem efetiva para apresentar modelos legíveis, mostrando atividades agregadas e omitindo detalhes irrelevantes, este trabalho propõe a criação de mapas declarativos hierárquicos independentes da linguagem de modelagem, utilizando uma hierarquia linguística de atividades. A abordagem proposta aplica Processamento de Linguagem Natural para a construção de modelos declarativos mais abstratos produzidos através da mineração de processos. Para isso, as relações semânticas hiperonímia e holonímia são aplicadas para a descoberta de hierarquias entre as palavras presentes nos rótulos das atividades de um processo. O método apresentado foi avaliado em um estudo de caso com dados de um processo real e com suporte de especialistas no domínio. Os resultados mostraram que é possível gerar grupos de atividades significativos analisando os rótulos das atividades, de maneira a possibilitar a criação de visões mais abstratas e com complexidade reduzida de um processo, partindo de um mapa declarativo de nível mais detalhado.

Palavras-chave: Mineração de Processos, Modelagem Declarativa, Abstração Semântica.

Table of Contents

List of Figures

List of Tables

Chapter 1 - Introduction

This chapter presents the main aspects of this research, including its motivating factors, the problem characterization, the hypothesis to be investigated and a solution proposal. In addition, the research methodology and thesis structure are presented.

1.1 Motivation and Problem Characterization

Business Process Management (BPM) includes methods, techniques and tools to support the design, enactment, management, and analysis of operational business processes (VAN DER AALST et al, 2003). Besides the traditional use of process models within software engineering, they are more and more used for pure organizational purposes. Within BPM, process modeling is supposed to be an instrument for coping with the complexity of process planning and control (BECKER et al, 2000).

A process modeling language must provide concepts for representing processes. An imperative process modeling language focuses on the aspect of continuous changes of the process' objects, where each object's life can be described in terms of a state space that formulates its possibilities to get from one location to another, by the so-called state changes. In addition, its transition space formulates how distinct actions, events and changes can possibly succeed each other during the execution of a business process (FAHLAND et al., 2009). Imperative process modeling is characterized by an inside-to-outside approach. It primarily specifies the procedure of how work has to be done (PICHLER et al., 2012).

A declarative process modeling language, by contrast, focuses on the logic that governs the interplay of actions and objects of a business process. Its concepts describe the key qualities of distinct objects and actions, and how they relate to each other in time and space. These relations may be arbitrary and do not need to be continuous. A declarative language is insensitive to how a process works; rather, it aims to describe what the essential characteristics of a business process are (FAHLAND et al., 2009). Declarative process modeling is referred to as an outside-to-inside approach. In contrast to imperative languages, declarative languages do not specify the procedure a priori (PICHLER et al, 2012).

Process mining techniques allow knowledge extraction from events stored by information systems. It allows the automatic creation of process models by analyzing data from previous process executions stored in software systems, what should be infeasible for doing manually depending on the amount of data involved. They are also an important connection between data mining and business process management. The interest on this topic has grown due to the advancement on computers technology and processes management, so even more events can be registered on event logs and more details about business process are available. In addition, there is a need for improving and supporting business processes in a competitive and rapid changing environment, as stated by VAN DER AALST et al. (2012). It is an emerging discipline which provides sets of tools to support fact-based insights and allow process improvements. The idea of process mining is to discover, monitor and improve real life processes by extracting knowledge from event logs readily available in contemporary information systems (VAN DER AALST, 2011).

The motivation for this work comes from the differences between imperative and declarative paradigms on process mining scenarios. Figure 1.1a shows a city railways diagram. There, one can see the paths where the trains can go through and it is known that they cannot move outside the rails. This is an imperative perspective, in which every possible behavior is made explicit. On the other hand, Figure 1.1b shows a vessel navigating through some nautical signs. We assume the ocean waterways are opened and, for security reasons, there is a need to avoid some dangerous places indicated by the signals. This is a declarative manner to describe the scenario, where the undesired behavior is constrained and the user is free to move around, just respecting the constraints defined.

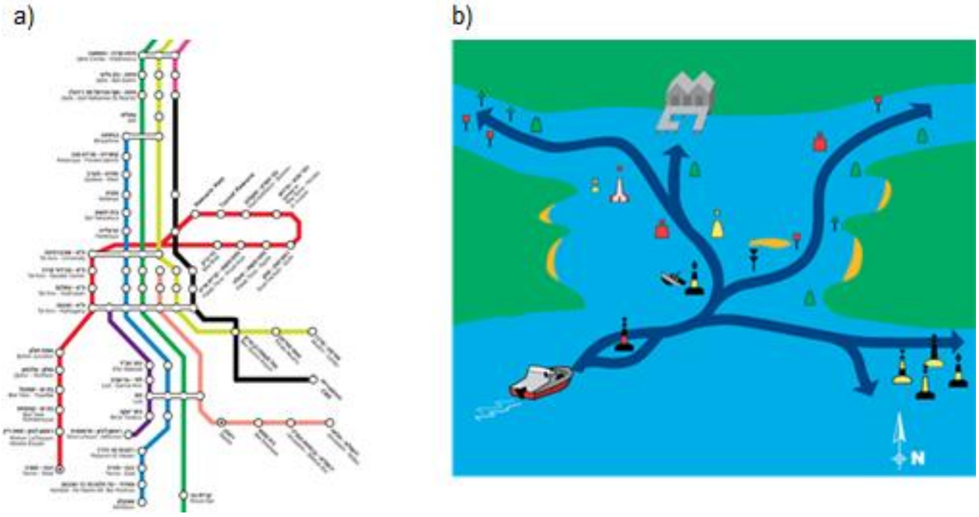


Figure 1.1: Imperative vs. Declarative paradigms.

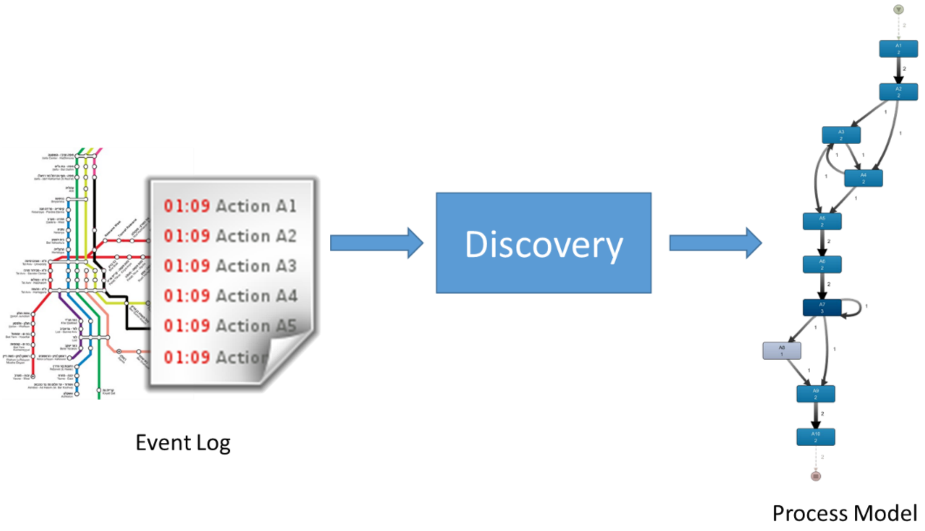


Figure 1.2: Model discovery of a well-structured process.

When analyzing a structured process (as in the railways example) with a well-known behavior, process mining has good chances to discover readable process models. This type of model displays all allowed paths, just as depicted in Figure 1.2.

However, in an unstructured process (as in the vessel example), there may exist multiple paths. When trying to discover a process model in the same way as presented before, there is a possibility to end up with a spaghetti-like model (Figure 1.3). This type of model is almost incomprehensible and does not help on understanding the behavior of a business process.

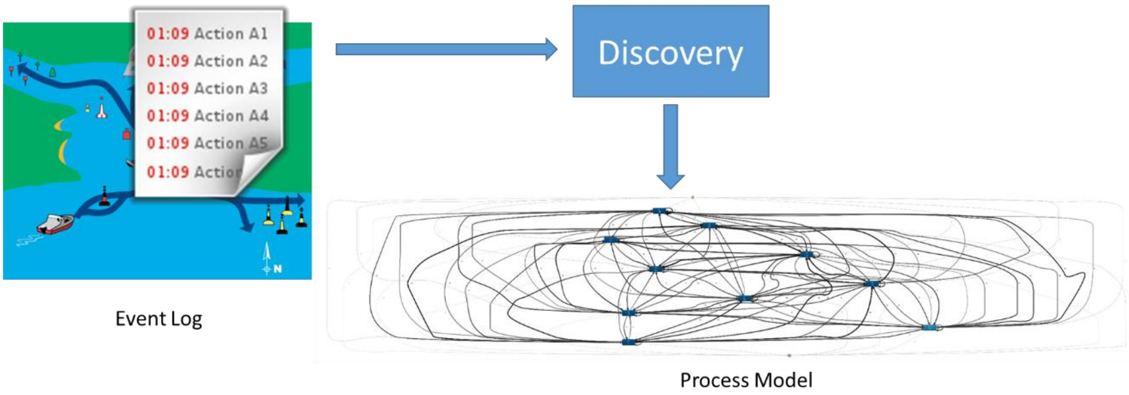


Figure 1.3: Model discovery of an unstructured process.

As shown above, automatically discovered models tend to be big and complex, especially on flexible scenarios, where process execution involves multiple alternatives. This behavior is common in domains such as medical treatments of patients, for example. In those scenarios, the information overload reduces model comprehensibility because traditional techniques used on discovery try to model every possible process behavior.

LA ROSA et al. (2011) said it is possible to relate process model complexity to its understandability. The authors state that an increase in size of a business process model may lead to comprehension problems. In addition, they state that for complex models it becomes difficult to validate, maintain and communicate them by its stakeholders. Thus, the understandability issues related to model complexity should be a concern for business analysts when creating and maintaining business process models.

According to FAHLAND et al. (2009), traditional process mining techniques usually represent discovered process using a so-called procedural or imperative modeling language, that is, one that focuses on the process control flow and the execution sequence of activities. Imperative models are appropriate to represent well-structured models, because they provide better support for analysis and execution direction. On the other side, unstructured processes need flexibility to drive changes and deviations on the activities flow. VAN DER AALST et al. (2009) show how a declarative approach enables a better balance between flexibility and support. While an imperative model describes exactly how a process must be executed, in a declarative way only essential model characteristics are described, thus requiring explicit definition of restrictions that limits process execution possibilities (REIJERS et al., 2013).

However, declarative process mining techniques may produce models with a high quantity of constraints, which may be incomprehensible for humans. Once dealing with flexible or unstructured models, the chance to obtain a model with this characteristic is higher. When a declarative process mining algorithm is used to discover a model from a flexible or unstructured process, it is expected to mine only constraints that limit process behavior. However, depending on the characteristics of the business process and the algorithms used, it is possible to end up with another spaghetti-like model, with lots of constraints (Figure 1.4). Also, the combination of constraints in a declarative process model might generate new hidden dependencies, which are complex and difficult to be identified by humans (HAISJACKL et al., 2013). REIJERS et al. (2013) said the increasing number of restrictions negatively affects the model quality. Even few activities with many restrictions can result in hard to understand models (MAGGI et al., 2012).

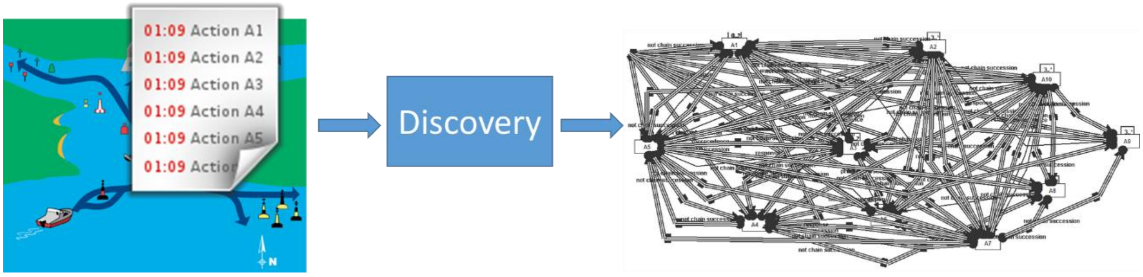


Figure 1.4: Declarative model discovery of an unstructured process.

This work addresses the problem of high complexity of declarative models generated by automatic process mining due the presence of a great number of discovered constraints that clutter a model.

1.2 Hypothesis and Solution Proposal

Considering the high complexity of declarative models created by process mining, and the possibility to improve the understanding of such class of process models, the hypothesis guiding this research is stated as:

IF semantics is used to find abstraction and aggregation relations between activity labels in a declarative model THEN it will be possible to generate groups of activity labels, with useful meanings for domain stakeholders, enabling the creation of abstractions over declarative process models in order to produce process views with reduced complexity.

From a business process perspective, subprocesses in declarative models behave differently than in procedural models: on imperative models, every process fragment ranging from a single entry and a single exit (SESE) can be grouped as a complex activity (WEBER et al., 2011). On declarative models, this structure is not informative enough, because the activities' sequence is not rigid, and the structural grouping of activities is inadequate. For declarative models, it should consider a common objective of the grouped activities (ZUGAL et al., 2013). The activities in a group should relate to a similar intention (SOFFER et al., 2012), making possible for a stakeholder to understand, in an more abstract view, that a group has a set of activities with a specific goal within the overall goal of a business process.

The use of modularization to hierarchically structure information was identified as a viable approach to deal with complexity for decades (HAISJACKL et al., 2013). Therefore, the present proposal intends to reduce declarative business process models complexity by automatically generating process hierarchies (defined by groups of activities), in which the proposed groups generalize lower level activities according to semantic relations. This kind of generalizations, i.e., abstractions and aggregations, are extensively used in database domains, and according to SMITH et al. (1977), generalizations are perhaps the most important mechanism for conceptualizing the real world. In addition, SMIRNOV et al. (2011) said that abstractions are seen as an effective approach to represent readable models, showing aggregated activities and hiding irrelevant details, and hierarchies may be used to perform aggregation, thus reducing the mental effort to understand a model (ZUGAL et al., 2013). While there exists a trade off between model size and the degree of hierarchy, it has been observed that, for larger models, hierarchy may have a positive influence on understandability (ZUGAL et al., 2013). Inspired by these previous works, the proposed solution is to retrieve semantic generalizations among activity labels in a process model to create groups of similar activities, and then use these groups to create more abstract views in declarative maps that should be less complex than the original low-level declarative map.

1.3 Research Goals

To cope with the high complexity of the automatically discovered declarative models, the present research proposes to create language-independent hierarchical declarative maps using a linguistic-driven hierarchy of activities, by grouping activities with common semantics instead of using process structure to create groups.

1.4 Scientific Method

The first step of the method consisted in conducting a bibliographic review in order to:

- Collect information about the state of art on process mining, process model abstraction and natural language processing (NLP) on business process models;
- Identify existing techniques to deal with complexity over discovered declarative process models;
- Delineate a rationale through which the proposed solution will increase the body of knowledge in business process management and reach the research goals.

The next step was the development of algorithms to handle process model's activity labels and then find semantic similarities using NLP. This step was followed by the implementation of the algorithms, and a prototype-running software, in order to run experiments to evaluate the solution. After several tests and tuning of algorithm's parameters, a case study was conducted with real life process execution data. An event log was used to generate a declarative map, and the proposed method was applied to create an abstraction layer over the map, aiming to produce a less complex model. This research proposal was evaluated through an explanatory approach (RECKER, 2012) and an experiment. In order to assess the usefulness of group suggestions by following the requirement of having activities in a group with similar intentions (ZUGAL, 2013), an online survey with domain specialists was performed. A quantitative analysis was conducted on the data gathered by the experiment and was evaluated using process models complexity metrics to confirm complexity reduction on the resulting declarative map with abstractions.

1.5 Document Structure

The remaining of this work is structured as follows: Chapter 2 presents theoretical background, where declarative modelling, process mining and natural language processing fundamentals are explained. Chapter 3 analyzes related work and its importance for this research. Chapter 4 details the proposed method and shows an example execution. Chapter 5 discusses a case study performed in a real world environment. Chapter 6 concludes this thesis with final considerations about its contributions, difficulties found during the development of the research and future works.

Chapter 2 - Theoretical Background

This chapter presents the theoretical background necessary to build the solution proposal. The fundamentals of business process management, process modeling, process mining and their interplay with natural language processing are discussed, with special emphasis on declarative models that are the focus of this research.

2.1 Business Process Management

“Business Process Management is the art and science of overseeing how work is performed in an organization to ensure consistent outcomes and to take advantage of improvement opportunities.” (DUMAS et al., 2013). Processes are everywhere in organizations, where they need to manage their own processes, even when they are not aware of them. Some typical processes are order-to-cash, procure-to-pay, issue-to-resolution and application-to-approval. Business processes are what organizations do to deliver a product or a service to their customers (DUMAS et al., 2013). From the management perspective, it is very important to know which processes are intended to be improved and how to do it. The BPM lifecycle (Figure 2.1) supports these tasks (VAN DER AALST, 2013) by structuring phases that direct business process management.

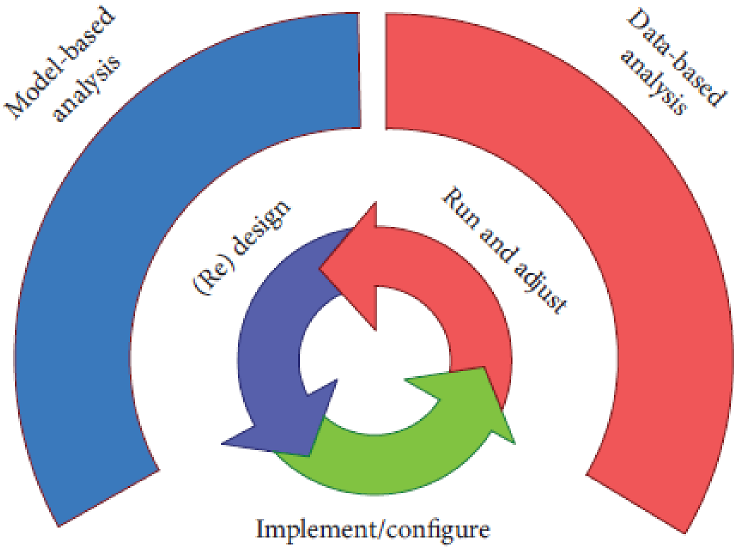


Figure 2.1: The BPM lifecycle and its three phases (VAN DER AALST, 2013).

The (Re)design phase consists on producing or updating a process model, that can be implemented or configured to the next phase. Afterwards, the process runs, and then it is enacted and adjusted if needed. In this phase (Run and Adjust) the process is not redesigned, only predefined controls are adjusted to adapt the process. The running process produces event data that can be collected by information systems to drive data-based analysis. This event data can be used to discover opportunities for improvement, such as bottlenecks, time waste and deviations. This is the input to the redesign phase, where process models are analyzed against the information produced by their own execution.

2.2 Knowledge Intensive Processes

Knowledge Intensive Processes (KIPs) are characterized by their operation's complexity (ABECKER et al, 2001), where each decision step is based on the experience of the people executing the process. In addition, they are considered people-centric processes, once they are performed by autonomous decision makers with different backgrounds, experience and expertise, also called knowledge workers (DAVENPORT, 2005). BAYER et al. (2006) states that KIPs are distinguished by their dynamic course of action where it is needed an environment with sufficient flexibility to support quick changes of resources and tasks involved on the process.

This type of process demands more interactions among process' participants, and as they commonly present a loose structure, people involved in a KIP need to communicate and use previous experiences for solving new issues and include innovation in their decisions, in order to reach the process goals (FRANÇA, 2012). MALDONADO (2008) states that a KIP is a semi or unstructured process with a high level of dynamic complexity, which is highly dependent on tacit and explicit knowledge of process' participants.

2.3 Business Process Modeling – Imperative versus declarative perspectives

Business process models are important at various stages of the BPM lifecycle, from the initial as-is model, passing through the execution, to the to-be model produced after the redesigned phase. There are many reasons for modeling a process, a simple but important one has great importance: to understand the process and share this understanding among people involved with the process. The idea is to provide a better communication of the process behavior to its users and then make people able to identify and prevent issues (DUMAS et al., 2013). The increasing adoption of BPM due the need of manage and document business processes foster the use of process models. Process models can also be used in different scenarios, such as providing knowledge, analyzing and redesigning processes (DAVENPORT et al., 1990) or specifying system requirements (DUMAS et al., 2005).

Business processes may be represented in various ways, including by textual descriptions. However, as a means of communication, they need to be easily comprehended, and large textual information may become difficult to understand, and subject to misinterpretations. To avoid this problem, it is common to represent process models as diagrams. If all the stakeholders are aware of the adopted graphical notation, models may be more easily comprehended, with less risk of misunderstandings (DUMAS et al., 2013).

While an imperative perspective for process representation specifies exactly how things must be done, a declarative approach focuses on the logic that governs interactions between the actions of a process, describing what can be done, restricting only the undesired behavior (ZUGAL et al, 2013). Figure 2.2 depicts how a declarative approach based on constraints allows much more behavior than traditional approach, restricting only the undesired behavior.

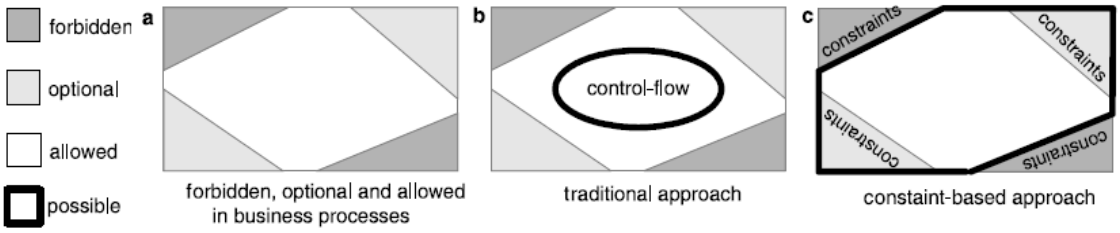


Figure 2.2: Different execution possibilities between imperative (traditional) and declarative (constraint-based) approaches (VAN DER AALST, 2009).2.3.1 Imperative Process Modeling Languages

There exist many languages to model imperative processes diagrammatically. The most basic are the flowcharts, consisting of rectangles to represent activities, and diamonds representing decision points. Today, several modeling languages are available, such as:

- UML Activity Diagram – AD (OMG, 2005): aims to model process and flow in software systems;

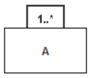
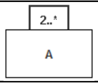
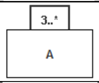
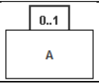
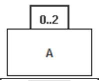
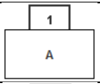
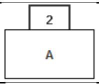
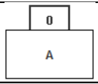
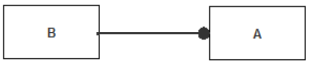
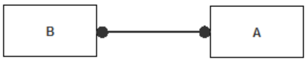
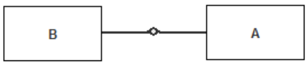
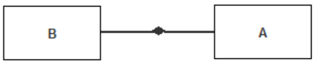
- Business Process Modeling Notation (BPMN) (OMG, 2011): developed for business processes modeling and execution. Its main concepts are similar to Activity diagrams;
- Event Driven Process Chain (EPC) (SCHEER, 1999): developed for business processes modeling. Its goal is to be easy to understand and to be used by business professionals. The basic elements are functions and events. Functions model activities from a process, while events are created by functions or external actors;
- Integrated Definition Method 3 (IDEF3) (MAYER et al., 1995): designed for modeling business processes and system flows. It has two perspectives: process scheme (process sequence model) and object scheme (object model and states transitions in a given process);
- Petri Nets (PETERSON, 1981): developed for modeling, analysis and simulation of dynamic systems with concurrency and non-deterministic procedures. They are used to model workflows;
- Role Activity Diagram (RAD) (HUCKVALE & OULD, 1995): presents roles, activities and interactions, also with external events.

All these languages have in common a procedural or imperative nature. That is, they require all execution alternatives to be explicitly specified in the model before the execution of the process. All new alternatives must be added to the model during design-time (PICHLER et al, 2012). If a behavior is not present in the model, it is considered forbidden.

2.3.1 Declarative Process Modeling Languages

An example of declarative modeling language is Declare (VAN DER AALST, 2009), which is grounded on restrictions modeled by linear temporal logic (LTL) expressions. LTL is a special type of logic that, in addition to the classical logical operators, also has temporal operators, such as always (□), eventually (◊), until (U) and next time (O). In Declare, templates represent a set of LTL expressions. These templates define relations between activities in a process model. A set of Declare constraints is presented in tables 2.1 (a) and (b). As an example, the LTL expression !((◊A)∧(◊B)) corresponds to the *not-coexistence(A,B)* Declare constraint template.

Table 2.1a: Existence and Choice Declare Constraints.

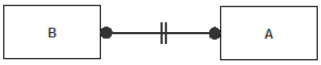
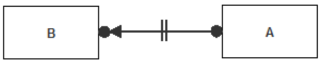
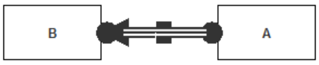
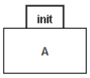
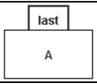
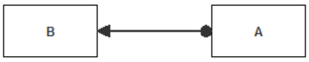
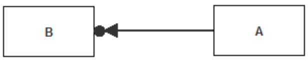
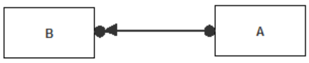
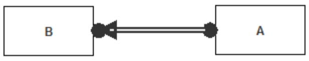
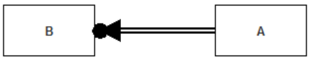
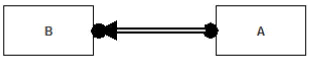
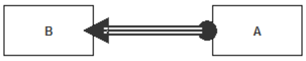
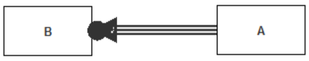
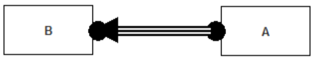
TYPE	CONSTRAINT TEMPLATE	MEANING	NOTATION
Existence	existence(A)	activity A has to be executed at least once.	
	existence2(A)	activity A has to be executed at lest two times.	
	existence3(A)	A has to happen at least three times.	
	absence2(A)	A can happen at most once.	
	absence3(A)	A can happen at most two times.	
	exactly1(A)	A has to happen exactly once.	
	exactly2(A)	A has to happen exactly two times.	
	absence(A)	A can never be started	
	responded existence(A,B)	If A happen (at least once) then B has to have (at least once) happpened before or has to happen after A.	
	co-existence(A,B)	If A happen (at least once) then B has to have (at least once) happpened before of has to happen after A. And vice versa.	
Choice	choice(A,B)	At least one from A and B has to be executed.	
	exclusive choice(A,B)	A or B has to happen but not both.	

Another approach for declarative process modeling is the use of Dynamic Condition Response Graphs (DCR Graphs), proposed by HILDEBRANDT et al (2010). A DCR Graph consists of activities, relations and runtime marking. Rectangles with an “ear” represent activities, and the “ear” contains the roles, which can execute the activity. The relations are drawn as arrows between activities with specific signs depending on the relation type. It is possible to nest activities under super-activities, in which case any relation that applies to the super-activity, applies to all its sub-activities.

Only atomic activities (that do not contain any sub-activities of their own) are executable (REIJERS et al., 2013). The core DCR Graphs model contains four binary relations between activities: dynamic inclusion, dynamic exclusion, condition and response relations (HILDEBRANDT et al. 2012). Table 2.2 shows these relations. A complete example of an execution of a DCR Graph is presented in (DEBOIS et al., 2014).

CARVALHO et al. (2013) proposed ReFlex, a graph based rule engine for the execution of declarative processes. The authors built a rule engine that is able to verify rules at runtime and control the execution of declarative processes. ReFlex has a graphical notation base on specialized edges that connect vertexes in a graph. Table 2.3 presents the edge types and their meaning and also its correspondence to Declare templates. Their graph-based engine claims that it is more efficient than traditional LTL-based verification systems, because they do not need to build a finite automata that grows exponentially in size for realistic models. However, the authors presented a limited set of rules that limits expressiveness when compared, for example to the Declare language.

Table 2.1b: Negation and Order Declare Constraints.

TYPE	CONSTRAINT TEMPLATE	MEANING	NOTATION
Negation	not co-existence(A,B)	Only one of the two tasks activity A or activity B can be executed, but not both.	
	not succession(A,B)	Before B there cannot be A and after A there cannot be B.	
	not chain succession(A,B)	A can never be executed directly after B.	
Order	init(A)	A is the first activity to be executed.	
	last(A)	A is the last activity to be executed.	
	response(A,B)	Whenever activity activity A is executed, activity B has to be eventually executed afterwards.	
	precedence(A,B)	Activity B has to be preceded by activity A. Activity B can happen only after activity A had happened.	
	succession(A,B)	Response and precedence together.	
	alternate response(A,B)	After each A is executed at least one B is executed. Another A can be executed again only after the first B.	
	alternate precedence(A,B)	B cannot happen before A. After it happens, it can not happen before the next A again.	
	alternate succession(A,B)	Alternate response and alternate precedence together.	
	chain response(A,B)	After A the next one has to be B.	
	chain precedence(A,B)	B can be executed only directly after A.	
	chain succession(A,B)	A and B can happen only next to each other.	

Both languages have tool support for design, enactment and execution of models, but only Declare offers support for analysis of Declare maps through the ProM[1] framework and support for process mining through the Declare Miner (MAGGI et al, 2011).

[1] For ProM refer to: http://www.processmining.org/prom/start.

For the scope of this work, both languages have sufficient expressiveness to be adopted for our proposed approach.

Table 2.2: DCR Graphs relations.

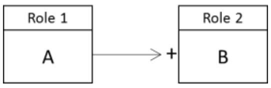
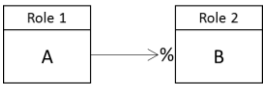
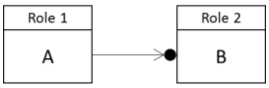
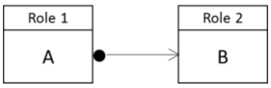
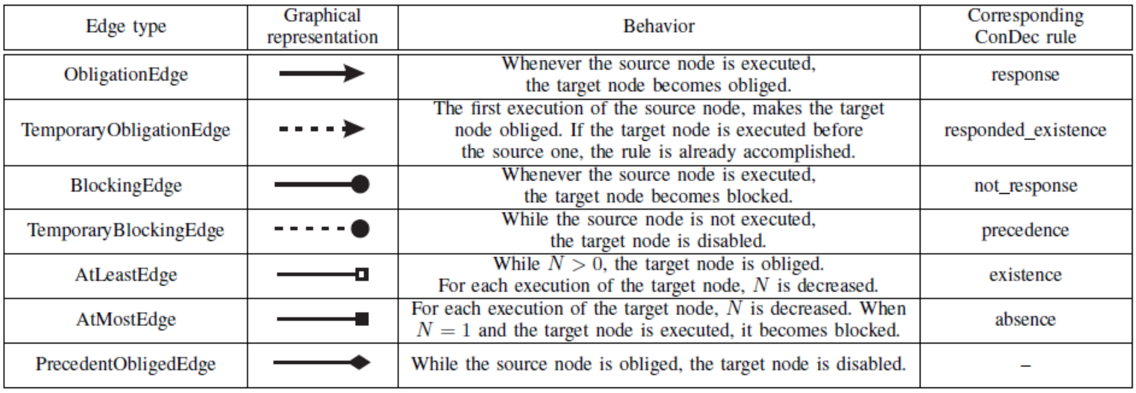
RELATION	MEANING	NOTATION
dynamic inclusion	The relation expresses that, whenever event A happens, it will include B in the graph	
dynamic exclusion	The relation expresses that, whenever event A happens, it will exclude B in the graph	
condition	If an event B has event A as condition, then event A must either be currently excluded or have happened for B to happen.	
response	If an event B is a response to an event A then B must happen at some point after event A happens	

Table 2.3: ReFlex's list of edge types and their representation (CARVALHO et al., 2013).



Edge type	Graphical representation	Behavior	Corresponding ConDec rule
ObligationEdge	──▶	Whenever the source node is executed, the target node becomes obliged.	response
TemporaryObligationEdge	- - - -▶	The first execution of the source node, makes the target node obliged. If the target node is executed before the source one, the rule is already accomplished.	responded_existence
BlockingEdge	──●	Whenever the source node is executed, the target node becomes blocked.	not_response
TemporaryBlockingEdge	- - - - -●	While the source node is not executed, the target node is disabled.	precedence
AtLeastEdge	──□	While $N > 0$, the target node is obliged. For each execution of the target node, N is decreased.	existence
AtMostEdge	──■	For each execution of the target node, N is decreased. When $N = 1$ and the target node is executed, it becomes blocked.	absence
PrecedentObligedEdge	──◆	While the source node is obliged, the target node is disabled.	–

In general, a declarative process model P = (*A*, *C*, *Λ)* can be defined as a triple consisting of: *A*, a finite non-empty set of activities; *C*, a finite set of constraints; and *Λ* a finite non-empty set of concepts which describe the constraints of a declarative language.

2.4 Process Mining

Process mining provides means to improve processes in a variety of applications domains (VAN DER AALST, 2011) due to two main reasons: First, more and more events are being recorded every day, providing detailed information about the history of processes, in a way never did before. Second, although Business Process Management and Business Intelligence software promise miracles, they did not completely fulfill the expectations from academics, consultants and vendors.

Process mining is considered a discipline, with aims to provide fact-based insights and to support process improvements. This discipline can be situated between computational intelligence and data mining on one hand, and process modeling and analysis on the other hand (VAN DER AALST et al., 2012).

2.4.1 Fundamentals of Process Mining

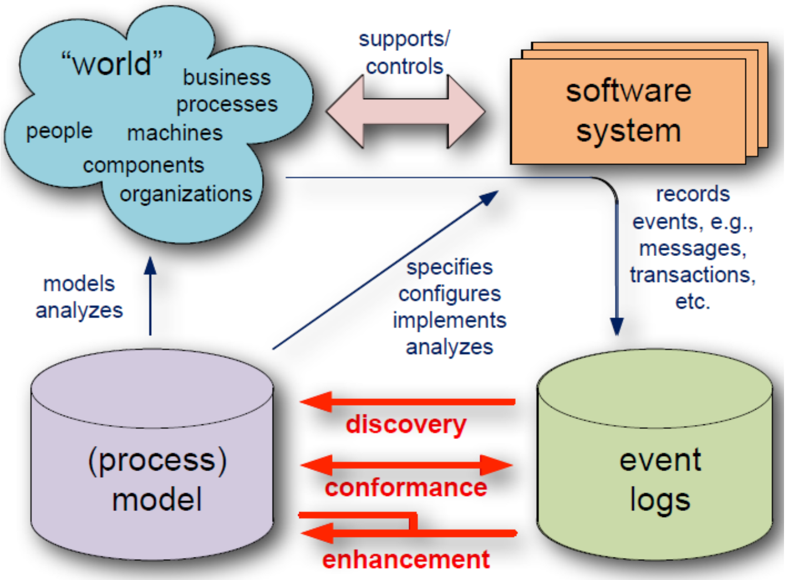


Figure 2.3: Process mining types and relationships (VAN DER AALST et al., 2012).

Figure 2.3 depicts how process mining relates to real world and software systems. Real world business processes may be supported or controlled by software systems. If these systems are capable to record the history of execution of their supported processes it will be possible to do process mining over their event logs. There are three types of process mining tasks (VAN DER AALST et al., 2012): discovery, conformance and enhancement. The first type is discovery, where an event log is taken and after processing, a process model is produced without using any previous information. The second type is conformance, where an existing process model can be compared with an event log of the same process. Conformance helps checking if reality (event log) conforms to a model and vice versa. The third type is enhancement, where an existing process model can be improved by using the information obtained for several real world executions of the processes, e.g. bottlenecks can be solved, the mainstream flow can be optimized, and services level agreement can be updated. As the interest of this work is on discovery, it is important to consider the following definition regarding the process discovery task (VAN DER AALST, 2012). A *process discovery algorithm* is a function that maps an event log L onto a process model P such that model is representative for the behavior seen in the event log.

There are many process mining techniques and tools, from both academic and commercial initiatives. Commercial examples are Fluxicon's Disco[2], Perceptive Software[3], and Celonis Process Mining[4]. From academia, most of process mining algorithms are implemented as plugins to ProM's framework. ProM provides an environment specially designed for developing process mining algorithms, covering all the three types of process mining.

2.4.2 Event Logs

Many current information systems are capable to log large amounts of events, especially due to the development of modern systems called Process-Aware Information Systems (PAIS). These kinds of systems are built with the idea of isolating the management of processes in a separate component from the remaining system's technological infrastructure. DUMAS et al. (2005) said that pulling away the process logic from application programs and capturing this logic in high-level models facilitates redesign and organic growth. A PAIS can be designed in a way to generate event logs that are well structured and provide detailed information about processes execution. Besides this, there is also a wide range of software that stores this information in unstructured form. In such cases, event data exist, but some efforts are needed to extract useful process information, before process mining (VAN DER AALST, 2011).

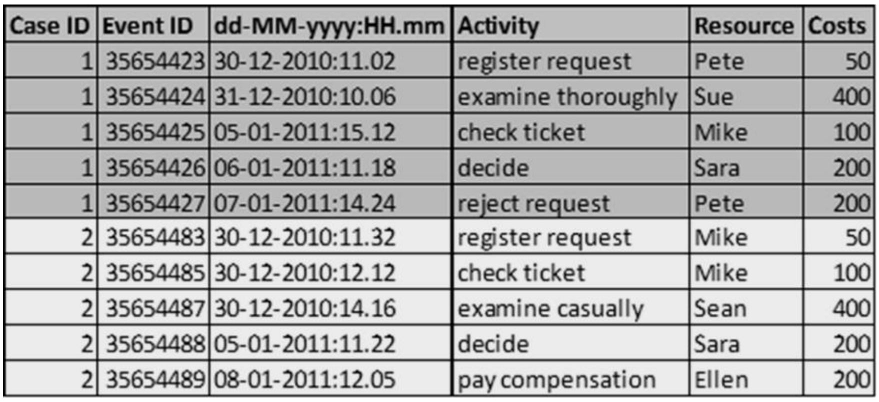
The basis for process mining is the existence of an event log. Event logs are characterized by a history of events over time, with specific characteristics. Table 2.4 shows typical information stored in an event log. From this table, each line represents an event. The "Case ID" field groups all events that occurred in the same process instance. There exists a unique identifier for an event occurred in a given timestamp, and the name of the executed activity is also presented. In addition, the resource information points to the user (or user role) who executed the activity, and the additional information (in this example, "Costs") provides more details about the circumstances in which the event was executed.

[2] For Disco refer to: http://fluxicon.com/.

[3] For Perceptive Software refer to: https://www.perceptivesoftware.com/products/perceptive-process/process-mining.html.

[4] For Celonis Process Mining refer to: http://www.celonis.de.

Table 2.4: Event log fragment example (VAN DER AALST, 2011).



Case ID	Event ID	dd-MM-yyyy:HH.mm	Activity	Resource	Costs
1	35654423	30-12-2010:11.02	register request	Pete	50
1	35654424	31-12-2010:10.06	examine thoroughly	Sue	400
1	35654425	05-01-2011:15.12	check ticket	Mike	100
1	35654426	06-01-2011:11.18	decide	Sara	200
1	35654427	07-01-2011:14.24	reject request	Pete	200
2	35654483	30-12-2010:11.32	register request	Mike	50
2	35654485	30-12-2010:12.12	check ticket	Mike	100
2	35654487	30-12-2010:14.16	examine casually	Sean	400
2	35654488	05-01-2011:11.22	decide	Sara	200
2	35654489	08-01-2011:12.05	pay compensation	Ellen	200

As stated before, event logs can be stored in various forms. A standardization initiative has been conducted in order to promote guidelines to develop systems aware of the importance of process mining, and then produce event logs with higher quality that can foster process mining initiatives and improve its results. An example of this effort is the Extensible Event Stream (XES)[5], which is an open XML-based standard for event logs. There also exists the Business Process Analytics Format (BPAF)[6] standard, which is another XML-based standard to allow for the easy aggregation of information to the process level. However, BPAF does not contain the aggregation structures present in XES meta-model (ZUR MUEHLEN et al., 2010). Actually, XES standard is supported by a variety of process mining tools used in this work, such as Disco and ProM, so this was the standard chosen for storing event logs in this thesis.

2.4.3 Mining Declarative Process Models

Most of the process mining algorithms produce traditional imperative process models. These techniques are well suited for structured process, where there are not many possible paths for process execution. Although some of these techniques can handle event logs from flexible or unstructured models, declarative modeling is proposed in order to provide a better balance between flexibility and guidance support from such models (VAN DER AALST, 2009). Besides doing declarative modeling, the possibility for mining declarative models has also arisen due its applicability on event logs from unstructured or flexible process models.

An approach to mine declarative process models is to verify each trace from a log with all the possible LTL expressions that represents Declare templates. For each satisfied

[5] For XES standard refer to: http://www.xes-standard.org/.

[6] For BPAF standard refer to: http://www.wfmc.org/standards/bpaf.

expression, the respective template is stored as an output of the discovery process (MAGGI et al., 2011). The set of discovered Declare templates forms a process model. An implementation for declarative process mining is the DeclareMiner (MAGGI et al., 2011), which is available as a ProM plugin. This plugin offers parameters that can be set to discover only the interesting constraints (MAGGI et al., 2012), namely Support and Confidence, Interest Factor (BRIN et al., 1997) and Conditional Probability Increment Ratio (WU et al, 2004). In this plugin, the Support (Equation 1) of a Declare constraint in an event log L is defined as the fraction of process instances in which the constraint *C* is satisfied. The Confidence (Equation 2) of a Declare constraint in an event log L is the ratio between the Support of the constraint and the Support of the antecedent part of the constraint, which is the part that activates the constraint but not necessarily satisfies it completely. For the complete satisfaction of a constraint, the second part, called consequent, also needs to be activated. Confidence measures might be misleading in scenarios where the support of either the antecedent or the consequent is equal to 1.0 (MAGGI et al., 2011). To deal with such scenarios the Interest Factor provides a stronger dependency between the antecedent and the consequent parts as shown in Equation (3). Finally, Equation 4 presents the conditional-probability increment ratio (CPIR) measure that assesses whether two activities *a* and *b* are positively or negatively related.

$$supp(C) = \frac{satisfiedInstances(L, C)}{instances(L)} \tag{1}$$

$$conf(C) = \frac{supp(C)}{supp(antecedent)} \tag{2}$$

$$InterestFactor(C) = \frac{supp(C)}{supp(antecedent)supp(consequent)} \tag{3}$$

$$CPIR(C) = \frac{supp(C) - supp(antecedent)supp(consequent)}{supp(antecedent)(1 - supp(consequent))} \tag{4}$$

DI CICCIO et al. (2013) also developed a declarative mining algorithm that uses a probabilistic approach to discover Declare constraints, called MINERful++.

WESTERGAARD et al. (2013) proposed the UnconstrainedMiner algorithm to mine declarative models based on discovering regular expressions from event logs. In this proposal, each Declare template is translated to a regular expression that can be verified against the event log and also presents support and confidence calculations for each constraint discovered. In addition, some reduction techniques are applied to boost algorithm's performance, such as symmetry reduction, prefix sharing, parallelization, and super-scalarity. Symmetry reduction aims to discovering only one constraint that present any symmetry, which means that swapping some parameters does not change the validity of the constraint, such as *co-existence(A, B)* and *co-existence(B, A)*. For prefix sharing, when a log stems from a single process, the individual traces typically share a lot of characteristics which can be exploited to achieve improved performance. The idea is to organize the log in a prefix-tree or trie. This way it is possible to replay such a trie on an automaton in a single run, allowing to not replaying any shared prefix more than once. Parallelization means the possibility to mine constraints in parallel by splitting up the check of traces. Finally, super-scalarity allows mining multiple constraints at a time, much how like super-scalar processors execute different parts of multiple instructions at a time.

A performance comparison between DeclareMiner, MINERful++ and UnconstrainedMiner (WESTERGAARD et al., 2013) showed that UnconstrainedMiner presented a better performance and is capable of mining all Declare constraint templates. As a disadvantage, UnconstrainedMiner produces a raw textual-based result with all possible Declare constraints from an event log. This result needs to be filtered, improved and then converted to the graphical notation in order to obtain a more useful Declare map.

2.4.4 Techniques for Declarative Models Improvement

Some researchers explored techniques to improve declarative models discovery that usually results in models with a high quantity of constraints. BOSE et al. (2013) proposed a method to correlate process information data with Declare templates to obtain models with the most interesting constraints.

MAGGI et al. (2013) presented a set of techniques to discover and repair models to produce fewer but more surprising constraints. The first technique is to prune weaker constraints implied by stronger ones. DI CICCIO et al. (2013) pointed that Declare constraints can be structured in a hierarchy that shows subsumption relations between constraint templates (Figure 2.4).

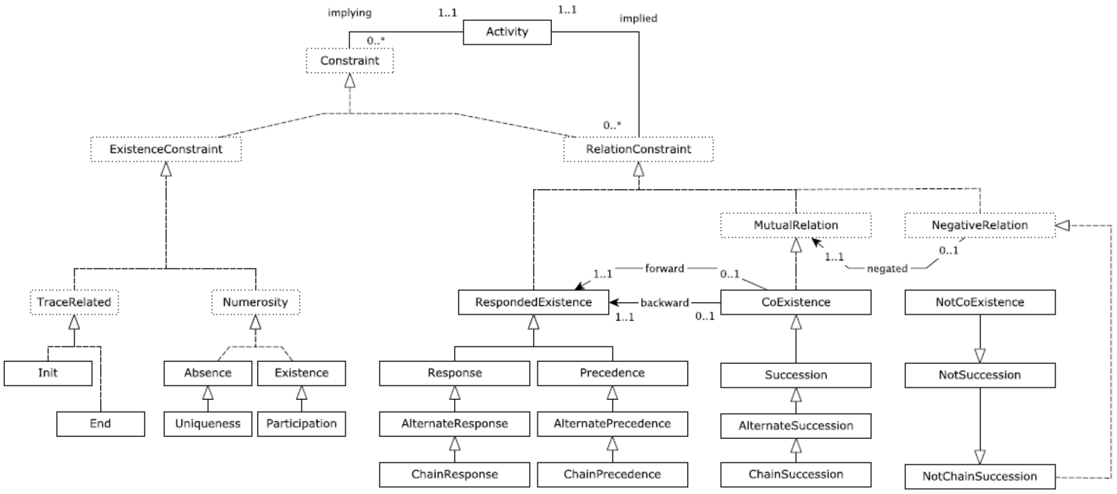


Figure 2.4: The declarative process model's hierarchy of constraints (DI CICCIO et al., 2013).

Based on generalization relations from Declare constraint's hierarchy, MAGGI et al. (2013) proposed to remove Declare constraints present in a declare map if there exists any declare constraint that subsumes the previous constraints. Figure 2.5 depicts the stronger-weaker relations that can be used to remove less interesting constraints.

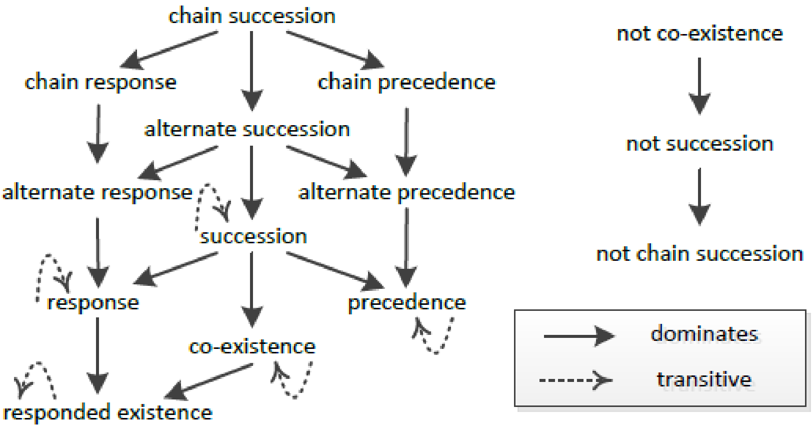


Figure 2.5: Dominating hierarchy of Declare constraints (MAGGI et al., 2013).

The second technique is the transitive reduction of Declare Maps. The interplay of three or more Declare constraints may imply on redundant constraints. Figure 2.6 shows an example of transitive reduction on the presence of three co-existence constraints connecting three activities. In this case just two constraints are necessary to model the desired behavior, thus, one constraint can be removed. Not all constraints can be pruned using transitive reduction. The dashed lines shown in Figure 2.5 point where transitive reduction can be applied.

In addition, a non-exhaustive set of reduction rules is presented that can be used on model's simplification. They also propose techniques for repairing Declare maps, such as strengthening constraints or removing constraints that no longer hold when compared to an event log, and the use of domain knowledge, e.g. using an ontology, to group activities based on their functionality. From this grouping, two classes of constraints can be distinguished: inter-group and intra-group constraints. Intra-group refers to the class of constraints where the activities involved in a constraint all emanate from a single group (Figure 2.7a), and inter-group refers to the class of constraints where the activities involved in a constraint belong to two different groups (Figure 2.7b).

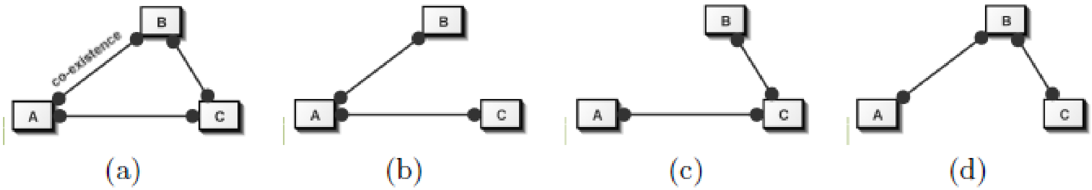


Figure 2.6: Transitive reduction for co-existence constraints: the original Declare map (a) can be pruned in three different ways using transitive reduction (b, c, and d) (MAGGI et al., 2013).

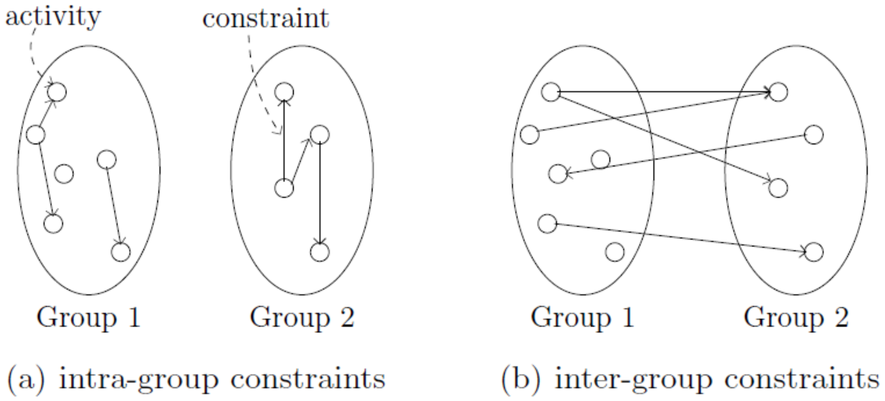


Figure 2.7: Inter- and intra-group constraints (MAGGI et al., 2013).

2.5 Complexity metrics for declarative process models

When dealing with conceptual models, the quality of a model may directly impact on the results of models usage. MOODY (2005) investigated more than fifty approaches that deal with quality of conceptual models. He pointed issues regarding the lack of empirical evaluations and accordance among terms and concepts. From the frameworks tested and studied by MOODY (2005), the proposal from LINDLAND et al. (1994) covers conceptual models in general. The fundamentals of this approach are related to three aspects: *semantics* or the relation between the model and the represented domain; *syntactics*, which is related to the modeling language and the evaluation of its consistency; and *pragmatics*, which relates the model to users participating on the creation and interpretation of the model. Thus, the pragmatic aspect goal is related to the comprehension of models. In line with this thesis' research problem, focus is given on improving user's understandability of a declarative model produced by process mining, which is a pragmatic aspect. In order to measure understandability, it is important to define objective criteria. This way, complexity metrics for business process models may be used to quantitatively define how easy a model is to be understood and maintained (GRUHN et al., 2006).

From software complexity, NAGAPPAN et al. (2006) point that there is no single set of complexity metrics that could act as a universally best defect predictor for software programs. In the same way, several process metrics can be designed to analyze business processes (CARDOSO et al., 2006) from different perspectives. The authors adapted some of the most well-known and widely used source code metrics to business processes. They also believe that the use of process metrics lies in using relatively simple metrics to build tools that will assist process analysts and designer in making design decisions.

Mechanisms for managing complexity of process models can be defined on two different levels: concrete and abstract syntax of a model. The concrete syntax deals with visual appearance, including symbols, colors and position, and is referred to as secondary notation. Abstract syntax of a process model relates to the formal structure of process model elements and their interrelationships (LA ROSA et al., 2011). The authors propose patterns to reduce the model complexity on the level of its abstract syntax, where each pattern that operates on a process model is related to a desired improvement of a structural metric. This work defines specific terms related to process models used in the complexity metrics. The term *module* indicates a process model which is part of a larger business process. The term *modular level*, to denote a hierarchical relation among modules which contain or are contained by another module. The term b*lock-structure* to denote a part of a model where each split element has a corresponding join element of the same type, and split-join pairs are properly nested. It is pointed that certain structural metrics can be related to process model understandability (MENDLING et al., 2007). LA ROSA et al.

(2011) discussed the following language independent metrics against the patterns for abstract syntax modifications:

- module size, the number of nodes in a *module*;
- model size, the summed size of all modules in a process model;
- repository size, the summed size of all models in a process model repository;
- models, the number of models in a process model repository;
- depth, the number of modular levels appearing in a process model;
- diameter, the longest path from a start to an end element in a process model;
- average gateway degree, the number of nodes a gateway in a specific process model is on average connected to;
- structuredness, the restructuring ratio of the number of nodes in an unstructured model to a block-structured variant of it (LAUE et al., 2010);
- modules overhead, the ratio between modules and model size;
- fan-in, the average number of references to a module;
- different modeling concepts, the number of different modeling concepts used in a process model.

With respect to the model characteristics, MENDLING et al. (2007) found that model size is of dominant importance on model's understandability.

For declarative models, some of the proposed metrics above are not applicable due the absence of a graph representation; hence, path related metrics should be discarded. In the same way, due the absence of control nodes, gateway related metrics are not suitable for declarative models.

In addition to the aforementioned complexity metrics, the number of constraints in the model may be used as a complexity metric for declarative models, because models with many restrictions, even with few activities, are considered hard to understand (MAGGI et al., 2012).

For the purpose of this work, considering the idea to use simple but meaningful metrics, the dominance of model size, and the impact of the quantity of constraints on declarative model's understanding, the following metrics will be considered: model size (splitted by number of activities and number of constraints), number of different modeling concepts, number of groups and constraint/activity ratio. To operationalize these metrics they can be defined as follows.

Number of Constraints (N_C). Let P be a declarative process model, containing a set of constraints $C \in P$. The number of constraints is the size of the set C, as $N_c = |C|$.

Number of Activities (N_A). Let P be a declarative process model, containing a set of activities $A \in P$. The number of activities is the size of the set A, as $N_A = |A|$.

Number of Different Modeling Concepts (N_D). Let P be a declarative process model and Λ the set of concepts which describe the elements of a declarative language, $D \in P$ is a set of different modeling concepts where $D \subset \Lambda$. The number of different modeling concepts is the size of the set D, as $N_D = |D|$.

Number of Groups (N_G). Let P be a declarative process model, containing a set of groups of activities $G \in P$. The number of groups is the size of the set G, as $N_G = |G|$.

Constraint-Activity Ratio ($R_{C\text{-}A}$). The constraint activity ratio is the number of constraints N_c divided by the number of activities N_A, defined as $R_{C\text{-}A} = N_c/ N_A$.

2.6 Abstraction and Aggregation on Business Process Models

Abstraction is seen as an effective approach to represent readable models, showing aggregated activities and hiding irrelevant details (SMIRNOV et al., 2011). Model abstraction may be used in diverse situations, such as when there are occurrences of activities with similar properties (e.g., roles and data objects) (SMIRNOV et al., 2011), or when there are relations between their activity labels that can be identified in a meronymy tree (SMIRNOV et al., 2010).

Adjusting granularity level of a model depends on the stakeholder. Top level management tends to appreciate coarse-grained process descriptions that allow fast and correct business decisions. On the other hand, employees who directly execute processes value fine-granular specifications. Thus, it might be often the case that a company ends up with maintaining several models of the same business process (POLYVYANYY et al., 2015).

While on imperative models every process fragment ranging from a single entry and a single exit (SESE) can be grouped as a subprocess (WEBER et al., 2011), on declarative models this structure is not informative enough, because the activities' sequence is not very rigid. Hierarchies can be used to perform information aggregation and to hide the details, so a group of activities may form a subprocess that can be represented by a complex activity, this way the mental effort to understand the model can be reduced

(ZUGAL et al., 2013). Moreover, hierarchical decomposition on imperative process models is viewed as a structural measure that may impact model understandability (ZUGAL et al., 2011), but does not influence semantics. From this point of view, hierarchy in declarative process models may have implications on semantics. For example, consider the following constraints in the same declarative map: *not succession*(A,B) and *precedence*(A,C), also activities B and C must be grouped and then represented by a complex activity D. There is no single constraint template available that can represent at the same time that A must not happen after D, and for D to be executed, A must be executed before. ZUGAL et al. (2013) state that hierarchy should be handled with care. This is due the positive effects of information hiding, i.e., the quantity reduction of activities and constraints presented in the model, and increased pattern recognition that promise gains in terms of understandability, and the negative effects of the integration of constraints and switching attention between sub-processes that may compromise the understandability.

2.7 Natural Language Processing

The main goal of Natural Language Processing (NLP) is to reach a higher level of natural language comprehension by using computers (KODRATOFF, 1999). It includes a wide range of techniques including text string manipulation to automatic document processing and linguistic analysis of its elements.

As this work applies NLP over language provided in text form, some fundamental concepts are important to natural language text analysis domain (LEOPOLD, 2013): A text Corpora is a corpus with large collection of texts. Corpora can be annotated with linguistic knowledge such as part of speech tags or parse trees. This annotation is a valuable source of information for grammatical analysis, providing insights about the frequency of words or the likelihood of a particular part of speech sequence.

Part-of-speech (POS) tagging is the process of assigning a part of speech to each word in a given text. Parts of speech are different lexical categories at word level. They include nouns (N), verbs (V), adjectives (ADJ), adverbs (ADV), prepositions (PREP), and determiners (DET). The input for a POS tagger algorithm is a set of text strings and a tag set. The output is a single tag for each word. For example, consider the sentence "*Process the customer data*" (LEOPOLD, 2013). A POS tagger should tag each word according to its respective part of speech, resulting in: *Process*/V *the*/DET *customer*/N *data*/N. There exists rule-based taggers based on a large set of manually defined rules to assign tags to words. In addition, there exist stochastic taggers, which use statistical methods to compute the best POS tag for a given word in a given context. For this, it is necessary to train a parser by using annotated Corpora.

WordNet[7] is a lexical database for the English Language. It organizes nouns, verbs, adjectives and adverbs into sets of synonyms, called synsets. The synsets are linked via semantic and lexical relationships. Thus, WordNet is able to represent a net of meaningful related words and concepts.

2.7.1 Semantics

Linguistics is concerned with the form and structure of language (LEOPOLD, 2013), and for written languages, it has three main branches: *morphology*, the study of structure of words; *syntax*: the study of the structural relationships between words; *semantics*: the study of meaning. As the main goal of this work is to propose groups of similar activities based on their meaning, focus will be given on *semantics*.

LEOPOLD (2013) said that the meaning of words could be discussed from a relational perspective. Words can be semantically related to other words in various ways. These sense relations are an important concept of lexical semantics and are considered a predominant issue in the context of natural language analysis. The most important sense relations are (LEOPOLD, 2013):

- Synonymy: words having the same meanings, e.g. to buy & to purchase;
- Homonymy: words with multiple unrelated meanings, e.g. to order / application;
- Polysemy: a word with multiple related meanings, e.g. to acquire / table;
- Hypernymy: type-of relationship between words. Denotes the super-name of the relationship, e.g. vegetable -> carrot;
- Hyponymy: type-of relationship between words Denotes the sub-name of the relationship, e.g. carrot -> vegetable;
- Holonymy: part-of relationship between words. Denotes the whole-name of the relationship, e.g. hand -> finger;
- Meronymy: part-of relationship between words. Denotes the part-name of the relationship, e.g. finger -> hand.

In the context of this work, the sense relations hypernymy and holonymy are especially useful for finding semantic hierarchies between words. Figure 2.8 illustrates how

[7] WordNet is available at http://wordnet.princeton.edu/.

Hypernyms provide a more general concept from a group of related concepts, and how Holonyms present the whole concept from a group of part concepts.

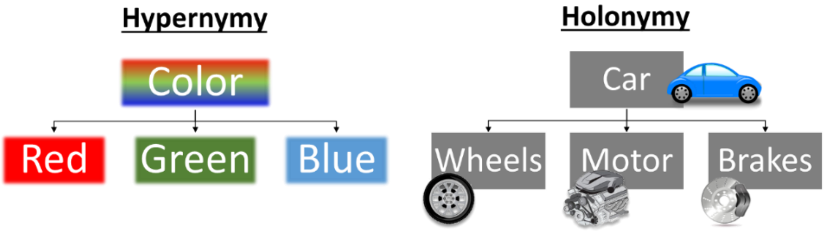


Figure 2.8: Hypernymy and Holonymy relationships.

2.7.2 Semantic Similarity Measures

Measures of semantic similarity based on WordNet have been widely used in Natural Language Processing. These measures rely on the structure of WordNet to produce a numeric score that quantifies the degree to which two concepts are similar. In their simplest form, these measures use path length to identify concepts that are physically close to each other and therefore considered more similar than concepts that are further apart (PEDERSEN, 2010).

RESNIK (1995) proposed to augment concepts in WordNet with Information Content values derived from text corpora. Information Content (IC) is a measure of specificity for a concept. Higher values are associated with more specific concepts (e.g., pitch fork), while those with lower values are more general (e.g., idea), and it is computed based on frequency count of concepts in a text corpus. Each occurrence of a more specific concept also implies the occurrence of the more general ancestor concepts (PEDERSEN, 2010). Information Content calculation is presented in Equation 5 and it is defined as the negative log of the probability of that concept, based on the observed frequency counts.

$$IC(c) = -logP(c) \tag{5}$$

Information Content can only be computed for nouns and verbs in WordNet, since these are the only parts of speech where concepts are organized in hierarchies. Since these hierarchies are separate, Information Content measures of similarity can only be applied to pairs of nouns or pairs of verbs (PEDERSEN, 2010).

There are many proposals that use the distance between two concepts in a taxonomy as basis for similarity. LIN (1998) presented a comparison between the most commonly used similarity measures, such as (RESNIK, 1995) and (WU & PALMER, 1994). As a result, it was demonstrated that Lin’s definition of similarity was the most similar to human judgments than the other two measures. Lin's similarity is defined in Equation 6. Given two concepts c_1 and c_2, their similarity is calculated by the division of twice the Information Content value of the least common subsumer (LCS) between c_1 and c_2, by the sum of the Information Content of c_1 and c_2. For example, consider the semantic similarity calculus between the concepts *cat* and *dog*. From a database with word frequencies, such as WordNet, the information content obtained for *cat* is *IC(cat)* = 8.63 and for *dog* is *IC(dog)* = 7.74. The least common subsumer between *cat* and *dog* is *carnivore*, and its information content is *IC(carnivore)*= 7.25. Applying these values on Equation 6, the resulting similarity $sim_{lin}(cat,dog)$ is 0.89.

$$sim_{lin}(c_1, c_2) = \frac{2 * IC(LCS(c_1, c_2))}{IC(c_1) + IC(c_2)} \quad (6)$$

2.8 Process Model Matching

The focus of process model matching is to find similarities on different process models. For this thesis, we are interested in measuring the similarity of activities in a single process model, and then suggest groups of semantically related activities. Some of the presented techniques for process model matching can be applied to identify similarities between labels in a single process model and are considered in the construction of the proposal of this work.

Process model matching refers to the creation of correspondences between activities on different process models (CAYOGLU et al., 2013). The two major challenges of process model matching are textual heterogeneity and differences in model granularity, i.e., different abstraction levels between models (WEIDLICH et al., 2013).

CAYOGLU et al. (2013) presented the results from a process model matching contest where two datasets were proposed to test the capabilities of seven different approaches to match business process models. As a main characteristic, almost all matchers use a pairwise comparison at activity level to find similarities, and then use these results to calculate a score of similarity between two process models. The evaluated matchers consider syntactic, semantic and structural perspectives with different approaches. The overall results showed that there is no clear winner against the proposed datasets, because they have different characteristics such as, the number of complex correspondences and the linguistic consistency. The evaluation considered precision, recall and f-measure as metrics for evaluation. Using a gold standard for each dataset, each computed activity match was classified as either true-positive (TP), true-negative (TN), false-positive (FP) or false-negative (FN). Based on this classification, precision (TP/(TP+FP)), recall (TP/(TP+FN)), and the f-measure, which is the harmonic mean of precision and recall (2*precision*recall/(precision+recall)) were calculated.

When focusing on f-measure, the RefMod-Mine/NSCM (CAYOGLU et al., 2013) approach yields the best result for one dataset and bag-of-words similarity (KLINKMÜLLER et al., 2013) performed best for the other dataset.

RefMod-Mine/NSCM proposes an N-Ary Semantic Cluster Matching, which uses a semantic similarity measure for pairwise node comparison. Their similarity measure is based on matching equal stems (PORTER, 1997) of words divided by the sum of all words in a pair of nodes/labels. This approach also considers the presence of antonyms and the occurrence of negation words, but do not consider any other complex semantic relations. A user-defined threshold is used to filter low score matching.

KLINKMÜLLER et al. (2013) proposed the bag-of-words similarity for model matching. The approach consist on a pairwise comparison between activities and calculate a similarity score for a pair, based on the sum of the maximum similarity of each word in an activity label to all other words in its counterparty label, divided by sum of all words in the activity pair. Different from the previous approach, the similarity measure between words may be any available, such as Levenshtein distance (LEVENSHTEIN, 1966), Lin's metric (LIN, 1998) and stem-based metrics like RefMod-Mine/NSCM's approach. This approach also proposes a second technique called label pruning that attempts to better capture activity labels with a strong difference in specificity, such as matching "rank application on scale of 1 to 10" to "rank case". The idea is to remove a number of words from the longest label, based on some criteria, like lowest similarity scores or even document frequency. Again, a user defined threshold is used to prune low scoring matches.

2.9 Final Remarks

Business Process Management allows organization to be aware of how work is performed and where they have opportunities for improvement. Business process models play an important role on this scenario, they are important means of communication among the stakeholders and may be also used to build systems to support the execution of work modelled in a process. Due the advancement on computers technology and the spreading on BPM thinking, more and more systems are used to support business processes today. In these cases, the information about process execution stored in software systems is very valuable, as it can be used for the analysis of the performed work, the conformance of the activities being executed and also improvement opportunities. All these analysis may be operationalized via Process Mining, which allows the discovery, conformance and enhancement of business process models by analyzing data produced from the execution of business processes. A process mining initiative may present some difficulties when dealing with unstructured process. Even using a declarative approach to deal with flexibility and a loose structure on process models, process mining may produce models that are too complex due the presence of a high quantity of constraints. Business process abstraction must be a way to deal with this complexity. Moreover, declarative models may not rely solely on structural information to be used for abstraction purposes, however, the analysis of its semantics may help on the creation of abstractions.

Several approaches apply Natural Language Processing on Business Process Models with different goals. Examples are construction of models, design support, construction of formal specifications, quality assurance, generation of text from process models and information elicitation (LEOPOLD, 2013). The present work proposes to use NLP to create abstractions and aggregations by using hypernymy and holonymy semantic relations, in order to reduce the complexity of Declarative process models. The proposed approach can be situated between construction of models and information elicitation categories.

The fundamentals on NLP techniques and the use of WordNet, due the availability of a semantic taxonomy for the semantic similarity measures, are very important to reach the goal to group process activities with common semantics. In addition, a comprehensive study on the specific characteristics of hierarchies and subprocesses for declarative models is needed in order to build a solution that fits the proposed complexity reduction.

Chapter 3 - Related Work

This chapter focuses on presenting works that proposed to describe hierarchies on imperative and declarative models, and the existing research that applies these hierarchy concepts on process mining scenarios.

3.1 Hierarchies on Business Process Models

LI et al. (2011) proposed an approach to create different model abstraction levels to reduce complexity and improve understandability of imperative process models. They proposed a two-phase approach where the first step is the preprocessing of an event log in order to create a mapping between the original alphabet of the event log and an abstract alphabet, which enables mining models with abstractions. The proposed mapping is based either on domain knowledge defined by the user or on common sequential execution patterns found in event logs. After discovering the patterns or using domain knowledge, the event log is transformed by replacing the identified patterns by an abstract activity. In addition, the low-level abstracted part is maintained in a sub-log for further detailing. The second step is the process discovery over the abstracted event log with Fuzzy Miner (GÜNTHER et al., 2007) algorithm. FuzzyMiner was adapted to work with maps in order to enable zooming in the abstract activities to view the fine granular part of the model. For declarative models, however, the single identification of sequential patterns is not enough to infer groups of activities that constitute a subprocess, because activities' sequence is not rigid for all types of constraints.

BOSE et al. (2012) enhanced LI et al. (2011) proposal and defined a taxonomy for abstractions that considers loops and conserved regions relative to sequences in event log traces. They presented the graphical representation of abstract activities and the implementation on a new enhanced version of Fuzzy Miner's plugin. The authors considered the discovery of tandem arrays (loop patterns) and maximal repeats (common subsequence of activities within a process instance or across process instances) to identify common execution patterns that can be turned into abstractions. Their approach is compliant with imperative models; for declarative models, however, structural information based on control flow sequences is not sufficient for creating abstractions and in some real life situations, there is no domain knowledge available. In a different approach, the proposal of this thesis is to use semantic information available in process activity labels to find similarities and then create abstractions for declarative models.

BAIER et al. (2013) presented a method to construct abstraction layers in process models by matching events and activities. The main goal of their method is to abstract an event log to the same abstraction level needed by the business using domain knowledge from existing process documentation. Their approach searches for potential relations between events and activities by comparing all business objects in two ways: a simple string match and the decomposition of the business objects into their smallest semantic components that are then compared each other. After mapping event classes to activity classes, they use a clustering schema to group activities based on timestamp information to calculate minimal distances between activities in a trace. This schema maps event instances surrounded by an event context to a high-level activity The approach assumes the availability of process and activities descriptions to match event class names from an event log, also, their semantic comparison do not explore word sense relations between words. Differently, this thesis proposal does not demand any external context beyond the activity labels and it uses word sense relations, such as hypernyms and holonyms, to find similarities between words.

ESHUIS et al. (2008) proposed the construction of process views. A process view, for example, may hide details of an internal process that are secret of irrelevant for the consumer. Their approach focuses on structured processes and allows different views of the process from the consumer and provider. To create aggregation it requires block-structure in order to maintain correctness. This approach creates correct abstractions, but as unstructured process models often do not present block-structure, it is not possible to build reliable abstractions from unstructured processes. For a declarative perspective, this approach is not adequate because there are constraints that connect activities, which cannot be identified by looking to a minimal temporal distance, such as co-existence and response constraints.

3.1.1 Hierarchies on Declarative Process Models

ZUGAL et al. (2013) examined the effects of hierarchy on declarative models and analyzed the pre-conditions to propose the definition of subprocesses on declarative models. As a result, they confirmed that the structural grouping of activities is inadequate and, for declarative models, it has to be done considering a common objective among activities. In addition, subprocess activities should be executed in isolation from their top-level process, that is, constraints on top level must not influence the subprocess execution and vice-versa. For example, consider the process in Figure 3.1a: the complex activity B can occur any time, due to response relation. However if A is executed, B will eventually occur after A, but activities C and D (in Figure 3.1b) will only be executed during the instantiation of B, that is, the execution of A does not trigger C or D until B is initiated.

ZUGAL et al. (2013) said that the definition of hierarchies on declarative models has to be performed in specific situations, since the transformation of hierarchical structures back to flat models is not always possible without changing the process structure and, possibly, its semantics. However, they also affirmed that this possible loss can be compensated by the expressiveness enhancement of a model.

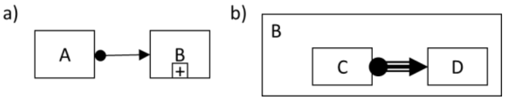


Figure 3.1: Example process with a complex activity B and its contained activities

DEBOIS et al. (2014) proposed a method for declarative modeling with subprocesses on DCR Graphs language. They defined a subprocess as a complex activity in the model which has underlying behavior instantiated when the subprocess is started. This approach allows single-instance or multi-instances of a subprocess, meaning that instances of the same subprocess can occur concurrently. In addition, an extension called Hi-DCR graph is proposed. This variant is equipped with a partitioning of its events into interface events and local events. It allows events to spawn subprocesses in a conservative way, where each instantiated subprocess is represented separately and the interfaces offer synchronization points among the subprocesses instances and the main process model. This approach does not try to aggregate constraints when representing subprocess, which should imply on semantic losses; instead, all relations between nested activities to other activities beyond the subprocess are made explicit. No constraint aggregation or subsumption is made, thus the original semantics of the low level model is preserved. Our approach follows the work of DEBOIS et al. (2014) with regard to creating abstract views from discovered Declare Models without causing semantic losses on the discovered processes models.

MAGGI et al. (2013) proposed the use of domain knowledge to model inter-group and intra-group constraints in declarative process model, after process mining. This paper do not directly propose the creation of abstractions, but emphasizes the improvement on understandability when separating constraint that are present inside groups of activities and other that relates different groups. Their approach assumes previous domain knowledge information to pre-define the groups. Our proposal extends this approach by automatically inferring groups from activity labels information and then creating a more abstract explicit representation of a declarative model.

MAGGI et al. (2014) proposed a technique for discovering a hybrid process model from an event log. A hybrid process model contains both imperative and declarative fragments structured in a hierarchy. Their discovery technique is sensitive to the nature of the traces and distinguishes structured and unstructured events. They conduct a context analysis considering predecessors and successors events. Procedural process mining is applied for the structured events, and declarative mining is applied for the unstructured. After mining these fragments, which should correspond to subprocesses, they are all abstracted in the main log and these steps can run iteratively again in order to build a hybrid hierarchy of activities. In order to decide which technique will be used to discover the main process, a string-edit distance (RISTAD et al., 1998) measure is applied over the abstracted event log. If the similarity among the traces is above 50% then procedural discovery is made, otherwise, declarative mining is used. Their approach focus on a novel approach to discover hybrid models, and to build hierarchies where the sequence of events is the main concern for splitting declarative and imperative fragments. No semantics are considered in their approach, and it will be an interesting opportunity to investigate if the declarative fragments discovered has some kind of a common objective among its grouped activities, as stated by ZUGAL et al., (2013).

3.2 Final Remarks

Regarding hierarchies, none of the above mentioned approaches addresses abstraction techniques on declarative process models to reduce their complexity. There exists several works dealing with hierarchies on imperative business process models, where structural and control flow aspects can be used to define subprocesses and abstract details. But, for declarative models, where block-structure is absent and flexibility is present, different techniques need to be employed to create abstractions. Existing works explain how to model hierarchy and subprocesses in declarative processes models, but fewer works applied these concepts on process mining scenarios, which sometimes result in very complex models and need some kind of automatic support for creating more abstract views that helps on model’s analysis. None of them focused on the semantics presented in activity labels to infer groups of similar activities. Although previous works did not completely solve the problem of the high complexity of automatically discovered declarative process models, they are very important knowledge sources for giving ideas on building the solution proposal presented in this thesis.

Chapter 4 - A Semantic Criterion to Create Abstractions for Business Process Activity Labels

This chapter presents the core contribution of this work, which is a method to create abstractions on business process models based on a semantic criterion, detailing all steps needed to reach this goal. In addition, a discussion about semantic similarity methods is made to allow comparison among the techniques and then discuss its advantages and disadvantages. Finally, to better illustrate the proposal, a complete example execution of the method is presented using an artificial event log.

4.1 Method Overview

The proposed method has the main goal to group similar business process activity labels to allow the creation of abstractions, represented by complex activities, which intends to create a more abstract process view from a declarative map resulting from a process mining step. Figure 4.1 shows an overview of the method.

The first step is the extraction of an Activity Labels List from an event log. Then a pairwise calculation of semantic similarity between each pair of activity labels is performed. The second step uses the semantic similarity values between each pair of activity labels to build a graph that allows grouping related activity labels defined by a cluster strategy based on maximal cliques on the graph. The output of this step is a set of groups of related activity labels. The third step consumes this last output, and also requires a declarative process model produced from the application of process mining techniques over the previously used event log. In this step, the activities on the declarative model are aggregated into complex activities, and the constraints between the aggregating elements are handled in order to create a more abstract view from the initial declarative model.

In the next sections the method is detailed and a step-by-step example is also presented.

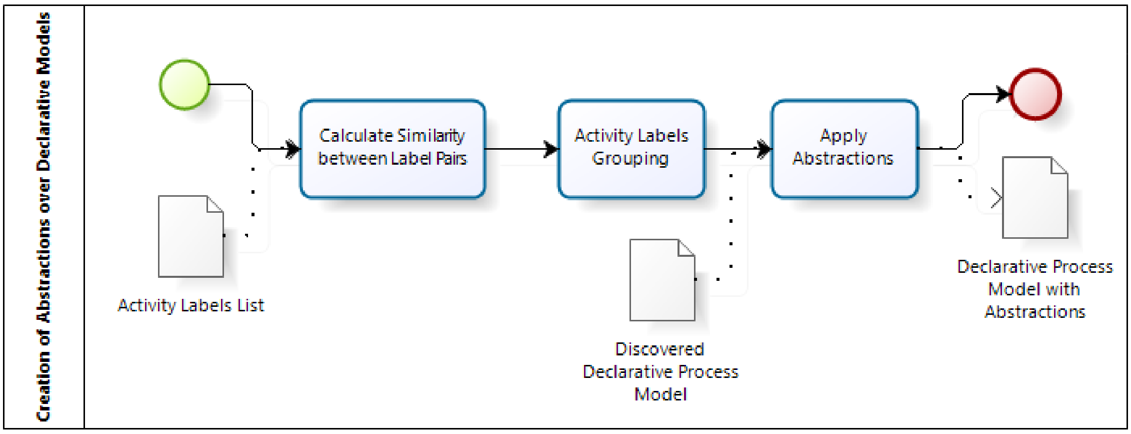

Creation of Abstractions over Declarative Models
Calculate Similarity between Label Pairs
Activity Labels Grouping
Apply Abstractions
Activity Labels List
Discovered Declarative Process Model
Declarative Process Model with Abstractions

Figure 4.1: Overview of the proposed method.

4.2 Calculate Similarity between Label Pairs

Inspired by the semantic approach of LEOPOLD et al. (2014) to name imperative process models and fragments, the proposed approach applies natural language processing to identify common objectives between activity labels, and then abstracts these activities creating hierarchies. More specifically, this work preprocesses an event log in which hierarchy (hypernyms) and inclusion (holonyms) semantic relations are used to group related activities as subprocesses. To create groups, the approach builds a graph with activity labels as nodes, and the similarity value between a pair of activity labels as edges. Algorithm 1 (Figure 4.3) calculates the semantic similarity between all pairs of activities, and is explained as follows.

For this method, Wordnet[8] was chosen to search for semantic relations. LEOPOLD et al. (2014) used a hypernymy and holonymy search scheme to abstract concepts from activity labels to suggest names for an entire process model or fragment. The proposed approach of this thesis aims to search for common objectives that can be used to gather activities in a subprocess by looking for common hypernyms and holonyms from each word belonging to a pair of activity labels (Step 3-7). As said by (PEDERSEN, 2010) it was considered only nouns and verbs from the labels, because these are the only parts of speech where concepts are organized in hierarchies in WordNet. Each word belonging to a pair of activity labels, being a noun or a verb, is called an origin word. To find the more abstract concept between two words, two approaches are proposed. The first searches for the most abstract concept for each origin word by traversing Wordnet's

[8] WordNet is available at http://wordnet.princeton.edu/.

hierarchical tree looking for hypernyms and holonyms relationships. A distance limit from the origin word should be defined to disconsider far concepts that may be too vague in Wordnet's tree. This leads to situations where two concepts that are in very different abstraction levels may not match a common concept. Nevertheless, choosing a broader limit would imply matching concepts that are close to the Wordnet's root (such as "entity"), even though there is not any interesting semantic relation. Figure 4.2a shows an example diagram of this search strategy for an activity label. Figure 4.2b presents an example result for the identified holonyms and hypernyms for the noun "method" with search limited by 2 levels from the origin word.

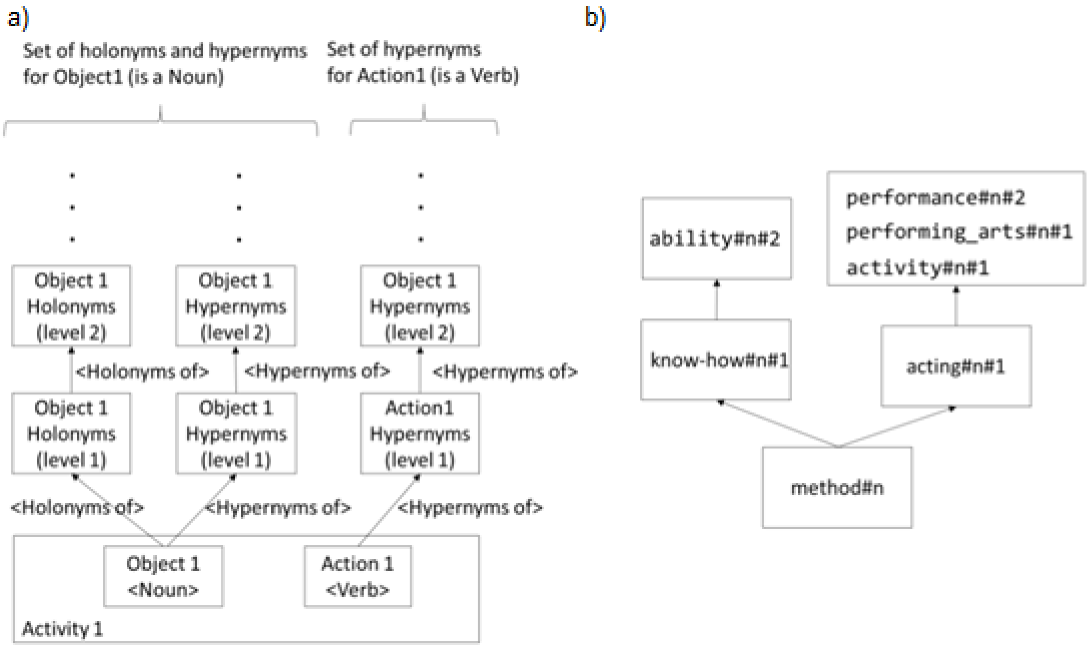

a)
Set of holonyms and hypernyms for Object1 (is a Noun)
Set of hypernyms for Action1 (is a Verb)
Object 1 Holonyms (level 2)
Object 1 Hypernyms (level 2)
Object 1 Hypernyms (level 2)
<Holonyms of>
<Hypernyms of>
<Hypernyms of>
Object 1 Holonyms (level 1)
Object 1 Hypernyms (level 1)
Action1 Hypernyms (level 1)
<Holonyms of>
<Hypernyms of>
<Hypernyms of>
Object 1 <Noun>
Action 1 <Verb>
Activity 1
b)
ability#n#2
performance#n#2
performing_arts#n#1
activity#n#1
know-how#n#1
acting#n#1
method#n

Figure 4.2: a) Identification of Hypernyms and Holonyms from words of an activity label. b) Example of identified holonyms and hypernyms for the noun "method".

For each possible pair of activity labels from the input list, the pairs of words of the same type (nouns to nouns, and verbs to verbs) are checked for common hypernyms or holonyms (if no common abstract concept is found, the matching between these two words is *null*) (Steps 12 and 17). When more than one abstract concept is found for the same pair of words, the most adequate abstract concept is selected by applying Word Sense Disambiguation (WSD) techniques, taking the set of all activity labels from the process as context information. The technique for WSD is WordNet::SenseRelate::WordToSet (MICHELIZZI et al., 2005), which aims to find the WordNet concept of a target word that is most related to a given set of words (Steps 13 and 18).

```
Algorithm 1: Calculate semantic similarity between activity labels
Input: List of unique activity labels $A$, number of levels to search in Wordnet's hypernymy and holonymy tree $k$
Output: Set of activity pairs with their respective average similarity measure $R$
Initialize $R$ with $\emptyset$
foreach activity label $a$ in $A$ do
    Apply part-of-speech tagging to identify all verbs $V$ and all nouns $N$ in $a$
    foreach verb $v$ in $V$ do
        Identify all hypernyms for $v$ until reach the $k$th level starting from $v$
    foreach noun $n$ in $N$ do
        Identify all hypernyms and holonyms for $n$ until reach the $k$th level starting from $n$
Generate a set $P_a$ with pairs of activities $p_a$(activity label $a1$, activity label $a2$) from the combination $\binom{A}{2}$
foreach activity label pair $p_a$ in $P_a$ do
    Generate a set $V_{1,2}$ with pairs of verbs $p_v(v_1,v_2)$ from the combination of each verb $v_1$ in $V_1$ from $a_1$ and each verb $v_2$ in $V_2$ from $a_2$
    foreach pair $p_v$ in $V_{1,2}$ do
        Match all common hypernyms $H_v$ between $v_1$ and $v_2$
        Invoke WordNet::SenseRelate::WordToSet algorithm to define the most adequate hypernymy $h_v$ from $H_v$, using $A$ as context
        Calculate Lin's semantic relatedness metric between $v_1$ and $h_v$ and $v_2$ and $h_v$
    Generate a set $N_{1,2}$ with pairs of nouns $p_n(n_1,n_2)$ from the combination of each noun $n_1$ in $N_1$ from $a_1$ and each noun $n_2$ in $N_2$ from $a_2$
    foreach pair $p_n$ in $N_{1,2}$ do
        Match all common hypernyms and holonyms $H_n$ between $n_1$ and $n_2$
        Invoke WordNet::SenseRelate::WordToSet algorithm to define the most adequate hypernymy or holonymy $h_n$ from $H_n$, using $A$ as context
        Calculate Lin's semantic relatedness metric between $n_1$ and $h_n$ and $n_2$ and $h_n$
    Calculate average semantic relatedness value $s$ considering all nouns in $N_1$, $N_2$ and verbs in $V_1$, $V_2$ to their most adequate hypernymy or holonymy
    Add $p_a$ and its $s$ value to $R$
return $R$
```

Figure 4.3: Algorithm to calculate semantic similarity between activity labels (RICHETTI et al, 2014a).

The second proposed approach is to look for the most similar parent concept (being a hypernymy or holonymy) between two words disregarding the number of levels on the semantic tree needed to find the more abstract concept indicated by the Common Parent Index (CPI). The Common Parent Index is the index of the node in the relationship that represents this divergence point. For example, when finding a hypernymy relationship between "dog" and "cat", the relationship is dog→canine→carnivore; cat→feline→carnivore, so "carnivore" is the word of the common parent index, and its index is 2 (FAN et al., 2010). As this approach tends to find concepts that can be too vague, a label pruning technique (KLINKMÜLLER et al., 2013) was used to balance labels with different sizes considering only a set of words with highest similarity.

Apart from the technique used, both approaches can be used as input for the next step, which calculates the semantic similarity between activity labels.

We also keep track of how strong a word is semantically related to its selected hypernymy or holonymy (Steps, 14 and 19). Semantic relatedness metrics can measure the degree of relationship between words, and the Lin metric was chosen because its results are similar to human judgment (LIN, 1998).

To operationalize the semantic similarity calculus between two activity labels, the following aspects should be considered. Given a process model P, we define an activity label $a \in A$, where A is the set of all activity labels from P. Moreover, a function $\omega: a \rightarrow W$ defines a set of words $w \in W$ assigned to an activity label a. For an activity label pair (a_1, a_2), a function $\delta: (a_1, a_2) \rightarrow D$ defines a set of word pairs $(w_1, w_2) \in D$ that are formed by combining each word from $\omega(a_1)$ with each word from $\omega(a_2)$, considering only noun-to-noun and verb-to-verb pairs (RICHETTI et al., 2014a). A function $\eta: (w_1, w_2) \rightarrow h$ defines the most similar hypernymy or holonymy h for both words, using any of the two previously presented approaches. Then, the semantic similarity between two activity labels a_1 and a_2 is calculated by summing the similarity values between each word w to its most similar hypernymy or holonymy h (defined by η) in a word pair and dividing this sum by two times the number of word pairs from D, as in Equation 7 (Step 20). Then the activity label pair and its similarity value is stored in a set (Step 21) and returned as output (Step 22).

$$\mathrm{sim}(a_1, a_2) = \frac{\sum_{(w_1,w_2)\in D}[sim_{Lin}(w_1, \eta(w_1, w_2)) + sim_{Lin}(w_2, \eta(w_1, w_2))]}{2 \times |D|} \tag{7}$$

As an example of the semantic similarity between activity labels, consider the following activity labels: “Receive Order” and “Pay Bill”. In this case there are two word pairs: *“Receive, Pay”* for the verbs, and *“Order, Bill”* for the nouns. Considering that the most similar hypernymy/holonymy for the verb’s pair is “*Get*” and for the noun’s pair is “*Legal document*”, the similarity calculation between these two activity labels is given by:

$sim_{Lin}(Receive, Get) = 1.00$

$sim_{Lin}(Pay, Get) = 0.00$

$sim_{Lin}(Order, Legal\ Document) = 0.79$

$sim_{Lin}(Bill, Legal\ Document) = 0.83$

$|D| = 2$

$$\mathrm{sim}(Receive\ Order, Pay\ Bill) = \frac{[1.00 + 0.00] + [0.79 + 0.83]}{2 \times 2} = 0.65$$

4.3 Activity Labels Grouping

This step groups activities into subprocesses, as illustrated in Algorithm 2 (Figure 4.4). It starts taking as input the output from Algorithm 1, the activity labels list and a user-defined minimum similarity value that is used to prune all pairs of activity labels with a similarity below this value (Step 2). Then, a list of pairs of activity labels that were not pruned is produced. This list contains candidate activities to form groups, and the activity label pairs are used to build an undirected weighted graph (Step 3), where each vertex relates to an activity label and the weighted edges represent the similarity value between two activity labels. The next step is to define how activities are grouped into a subprocess.

The grouping strategy finds maximal cliques using a graph representation (Step 5). Cliques are non-extendable groups such that each pair of nodes within a group has a relationship (BRON et al., 1973), i.e., a clique is a fully connected subgraph within a graph. For each clique, all its edges are summed (Step 7) and the clique with the highest sum is stored in a separate list (Steps 8-9), and its nodes are removed from the graph (Step 10). The algorithm then looks for the next clique with the highest sum in the graph, until there are no more edges in it. Figure 4.5 illustrates an example of our proposed grouping strategy.

From the example in Figure 4.2a, each node represents an activity label, while each edge represents the similarity between two activity labels. In Figure 4.2b, groups are formed by activities that are directly related to all other activities in a group, or a clique. From Figure 4.5c, a score representing the sum of edges from each group is calculated. The group with the highest score ({A,C,E}) is selected as a subprocess and removed from the graph. The group formed by the remaining nodes ({B,D}), which is also a clique, is selected as another subprocess (Figure 4.5d). The output from this phase is a set of activity label groups (Step 11), that can be used to create abstractions using the visualization proposal presented in section 4.6. If no groups suggestions are found, the original declarative process model will remain unchanged.

```
Algorithm 2: Group semantic related activity labels
Input: List of unique activity labels A, Set of activity pairs with their respective average similarity
       measure R, minimum similarity value v
Output: Set of activity labels groups S
Initialize S with ∅
Remove all activity pairs from R with average similarity measure below v
Create a undirected weighted graph G(V,E) where each vertex v is an activity label from A and
each edge e relates to a pair from R whose weight is the average similarity measure of the pair
while G has edges do
    Generate all possible vertex groups P where in a group each vertex relates to each other
    foreach group p in P do
        Sum the weight of all edges of p
    Identify the vertex group h with the highest weight sum
    Add h to S
    Remove all vertex in h from G
return S
```

Figure 4.4: Algorithm to group semantic related activity labels (RICHETTI et al, 2014a).

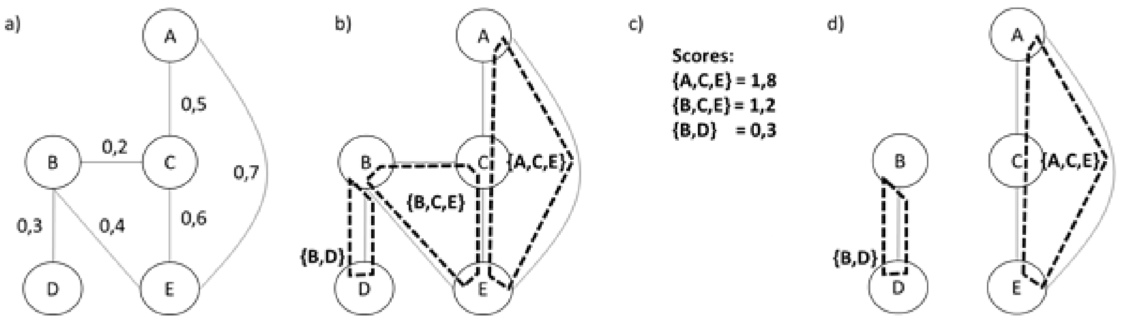


Figure 4.5: Proposed grouping strategy based o graph cliques.

4.4 Algorithms implementation

The proposed method takes as input a list of activity labels, and outputs a list of groups of activities. A flow diagram in Figure 4.6 shows all method’s steps implemented in a prototypical software. The prototype for executing Algorithms 1 and 2 was implemented in Java language. Auxiliary Python NLTK3.0[9] scripts were used for the part-of-speech tagging step. PERL Word-Net:SenseRelate::WordToSet[10] scripts were used to get the most adequate sense from a list of words to be disambiguated in a given context and Wordnet:Similarity[11] scripts provided the semantic similarity relatedness calculus.

[9] The toolkit is available at http://www.nltk.org/.

[10] Refer to http://search.cpan.org/~tpederse/WordNet-SenseRelate-WordToSet-0.04/.

[11] Refer to http://search.cpan.org/~tpederse/WordNet-Similarity-2.05/.

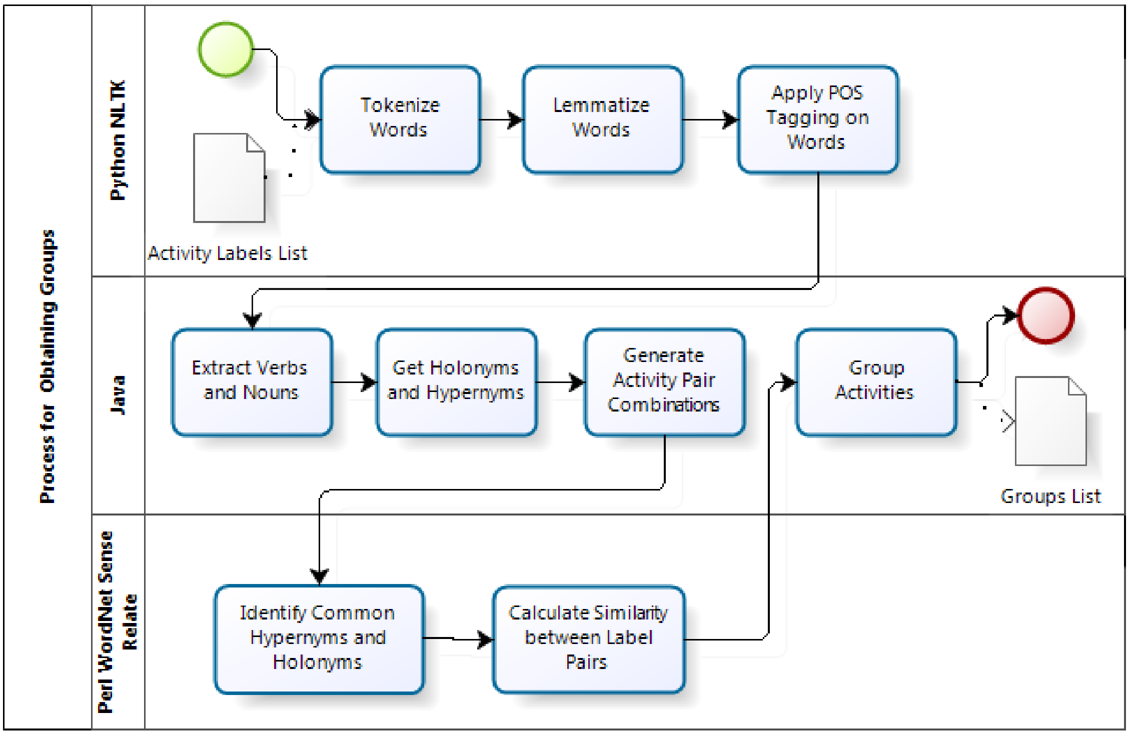

Process for Obtaining Groups
Python NLTK
Tokenize Words
Lemmatize Words
Apply POS Tagging on Words
Activity Labels List
Java
Extract Verbs and Nouns
Get Holonyms and Hypernyms
Generate Activity Pair Combinations
Group Activities
Groups List
Perl WordNet Sense Relate
Identify Common Hypernyms and Holonyms
Calculate Similarity between Label Pairs

Figure 4.6: Process implementation for obtaining the groups list.

The POS tagger is configured as an Unigram tagger trained with Penn Corpus[12]. The WordNet::SenseRelate[13] algorithm is used to get de most adequate sense from a list of words to be disambiguated and a given context. For the creation of groups after calculating the similarity between activity labels, the Bron-Kerbosch (BRON et al., 1973) algorithm was applied, which efficiently identifies cliques on graphs.

The similarity calculus variant, using the Common Parent Index, was implemented in Java using the Extend Java Wordnet Library (extJWNL)[14]. Considering the process from Figure 4.6, this variant perform the same activities and substitutes PERL's lane.

4.5 Evaluating the similarity measures for labels grouping

As stated in Section 4.1, many similarity measures can be applied in combination to the proposed grouping strategy. To look for the best similarity measure, four different techniques were analyzed:

[12] The corpus is available at: http://www.cis.upenn.edu/~treebank/.
[13] For Wordnet:SenseRelate refer to http://search.cpan.org/~tpederse/WordNet-SenseRelate-WordToSet-0.04/.
[14] For extJWNL refer to http://extjwnl.sourceforge.net/.

- hyho: hypernymy/holonymy match with a user defined search distance level, proposed in this work;
- hyhoCPIv2: hypernymy/holonymy match by looking for the common parent index, proposed in this work;
- BOWMax: an adaptation of the “Bag of words with label pruning” semantic similarity approach, which was defined in (KLINKMÜLLER, et al., 2013) for process model matching and modified to perform the label pruning by maintaining the words with the highest similarities;
- BOW2p: an adaptation of the “Bag of words with label pruning” semantic similarity approach, which was defined in (KLINKMÜLLER, et al., 2013) for process model matching and modified to perform the label pruning by considering their frequency on the process model.

To allow a quantitative evaluation, the following subprocess metrics defined by REIJERS et al. (2010) were adopted: precision, recall, f-measure, undershoot and overshoot. Precision (Equation 8) and recall (Equation 9) are common measures from the literature used to compare automated information retrieval to human information retrieval. In this case, the “retrieved information” to be compared is the groups of activities. This comparison, however, was performed as defined by REIJERS et al. (2010), who considered too strict to just compare exact matches of automated groups to human modeled groups. They argued that an automatically retrieved group with missing nodes when compared to its human-made counterpart is not a completely missed match, since in a real scenario the process analyst could easily investigate the subprocess and complete it. Therefore, they defined precision and recall on the activity level within groups or subprocesses, rather than in terms of exact matched groups. The author presents an example to justify this decision: suppose that the set of nodes {'receive request', 'fill out request', 'complete request', 'accept request', ‘file request'} constitutes a subprocess. If the set of nodes {'receive request', 'fill out request', 'complete request', 'accept request'} is returned automatically, then this is not an exact match, but it does provide useful information to the process analyst that is determining the subprocesses.

In addition to precision and recall, two metrics were proposed by REIJERS et al. (2010): overshoot (Equation 10) and undershoot (Equation 11). Overshoot is the fraction of found nodes on automated modeled groups that do not belong to any of the human modeled groups. For this two measures the lower values, the better. Undershoot is the fraction of nodes that do belong to human modeled groups but were not found in any automated modeled group.

Precision, recall, overshoot and undershoot are calculated according to equations 8, 9, 10 and 11, respectively, where p_M is a manually determined group, $\mathcal{P}_{\mathcal{M}}$ is the set of manually determined groups, p_A is an automatically determined group, and $\mathcal{P}_{\mathcal{A}}$ is the set of automatically determined groups. The function match(p_M) returns the most relevant automatically determined match for each manually determined group, or empty if no such match exist (REIJERS et al., 2010).

$$Precision = \frac{\sum_{p_M \in \mathcal{P}_{\mathcal{M}}} |p_M \cap match(p_M)|}{\sum_{p_A \in \mathcal{P}_{\mathcal{A}}} |p_A|} \tag{8}$$

$$Recall = \frac{\sum_{Pp_M \in \mathcal{P}_{\mathcal{M}}} |p_M \cap match(p_M)|}{\sum_{p_M \in \mathcal{P}_{\mathcal{M}}} |p_M|} \tag{9}$$

$$Overshoot = \frac{\sum_{p_M \in \mathcal{P}_{\mathcal{M}}} |match(p_M) - p_M|}{\sum_{p_A \in \mathcal{P}_{\mathcal{A}}} |p_A|} \tag{10}$$

$$Undershoot = \frac{\sum_{p_M \in \mathcal{P}_{\mathcal{M}}} |p_M - match(p_M)|}{\sum_{p_M \in \mathcal{P}_{\mathcal{M}}} |p_M|} \tag{11}$$

Also, the harmonic mean between recall and precision, called F-measure (Equation 12), was used as the main metric for starting the analysis:

$$Fmeasure = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{12}$$

To illustrate the calculation of subprocess metrics, consider an example situation where the algorithm suggest a single group with the activities “A, C and D”. In addition, there exists a manually defined group “A, B and C” to be used as a reference. By the comparison between the automatically and manually defined groups and by using the equations 8-12 it is possible to determine the subprocess metrics, as showed in Table 4.1.

Table 4.1: Example calculation of subprocess metrics for a single activity group.

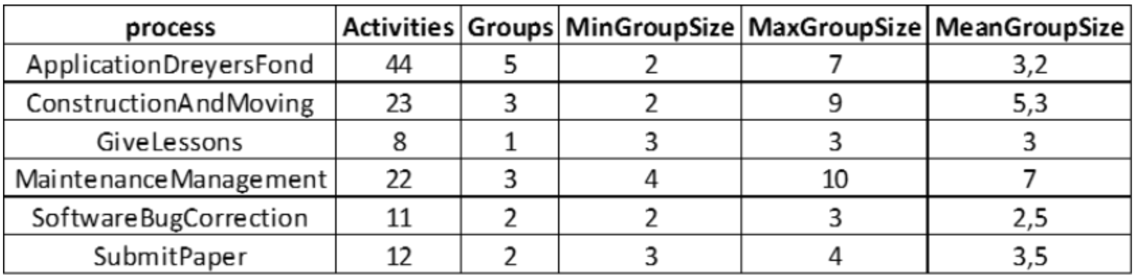
MANUAL GROUPING		A	B	C	
AUTOMATIC GROUPING		A		C	D
PRECISION	0,67	X		X	
RECALL	0,67	X		X	
OVERSHOOT	0,33				X
UNDERSHOOT	0,33		X		
F-MEASURE	0,67				

Auxiliary Values	
$\lvert p_A \rvert$	3
$\lvert p_M \rvert$	3
$match(p_M)$	2
$\lvert p_M \cap match(p_M) \rvert$	2
$\lvert p_M - match(p_M) \rvert$	1
$\lvert match(p_M) - p_M \rvert$	1

These subprocess quality metrics were implemented in Java to allow batch calculations varying the minimum similarity value, so it was possible to evaluate the best scenario for each technique.

The groups outputted by the proposed algorithms were compared to reference models with subprocesses (groups) built by analysts and available in the literature, namely the process “Writing a Scientific Paper” found in (ZUGAL et al., 2013), the processes “Bug Fixing Process of a Mobile App Company”, “How to Prepare Oneself and Materials for Teaching Pupils”, “Tasks at an Electronic Engineering Company” and “Looking for an Apartment and Buying It” found in (HAISJACKL et al, 2013), and the process “Application Dreyers Fond for ACM”[15]. Group hierarchies were previously manually defined for each of those models, and defined as reference standards to be compared to the subprocesses automatically defined using our approach. The process characteristics are shown in Table 4.1.

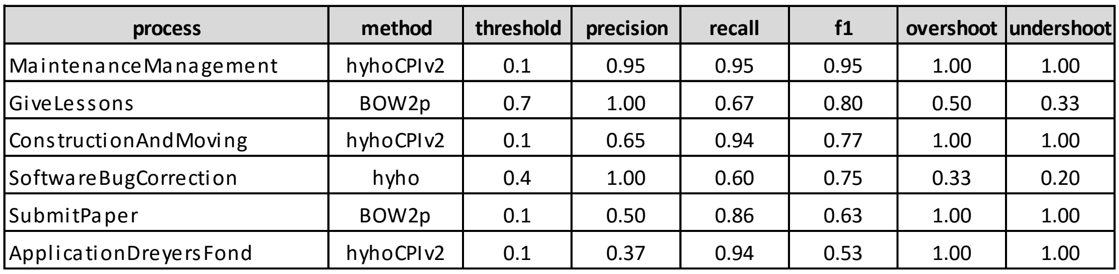
Table 4.2: Standard Process Models characteristics



process	Activities	Groups	MinGroupSize	MaxGroupSize	MeanGroupSize
ApplicationDreyersFond	44	5	2	7	3,2
ConstructionAndMoving	23	3	2	9	5,3
GiveLessons	8	1	3	3	3
MaintenanceManagement	22	3	4	10	7
SoftwareBugCorrection	11	2	2	3	2,5
SubmitPaper	12	2	3	4	3,5

The evaluation consisted on running each algorithm for each process model, with 24 combinations. In addition, the minimum similarity value was varied from 0.1 to 1.0 (with steps of 0.1) to tune the best similarity threshold scenario. This implied in 240 executions of the similarity calculus. The best result for each process model is shown in Table 4.2.

Table 4.3: Methods with best f-measure (f1) for each analyzed process model.



process	method	threshold	precision	recall	f1	overshoot	undershoot
MaintenanceManagement	hyhoCPIv2	0.1	0.95	0.95	0.95	1.00	1.00
GiveLessons	BOW2p	0.7	1.00	0.67	0.80	0.50	0.33
ConstructionAndMoving	hyhoCPIv2	0.1	0.65	0.94	0.77	1.00	1.00
SoftwareBugCorrection	hyho	0.4	1.00	0.60	0.75	0.33	0.20
SubmitPaper	BOW2p	0.1	0.50	0.86	0.63	1.00	1.00
ApplicationDreyersFond	hyhoCPIv2	0.1	0.37	0.94	0.53	1.00	1.00

[15] Model available at http://dcrgraphs.net.

Considering that the reference models have different characteristics, such as naming conventions, words quantity and manual grouping strategy of the modelers, some methods perform better than others depending on the scenario. The methods HyHoCPIv2 (3 in 6), BOW2p (2 in 6) and hyho (1 in 6) obtained the best f-measure score for at least one of the processes considered in this experiment. The BOWMax was the only method that did not outperform any other candidate for the analyzed scenarios. Even though showing the best results on f-measure, the methods HyHoCPIv2 and BOW2p in most of the cases they presented higher values on undershoot and overshoot, indicating that these methods produced groups that are less complete (higher undershoot) or with more activities than expected (higher overshoot) when compared to the reference standards. In practice, this results mean that the user may have to remove or include many activities in the resulting groups when applying these two methods, in order to make the groups correct.

The hyho method presented the lowest undershoot and overshoot values among all methods, in 5 out of the 6 scenarios. This result means hyho was able to produce groups with less missing or exceeding activities in comparison to other methods, although its maximum f-measure was not the highest one for the analyzed models.

From Figure 4.7, the best results for each method are presented to each selected process models. From these graphs it is possible to analyze the four subprocess metrics (precision, recall, overshoot and undershoot), one in each axis. In visual terms, the ideal geometric result would be a region delimited by the upper right quarter of the graph.

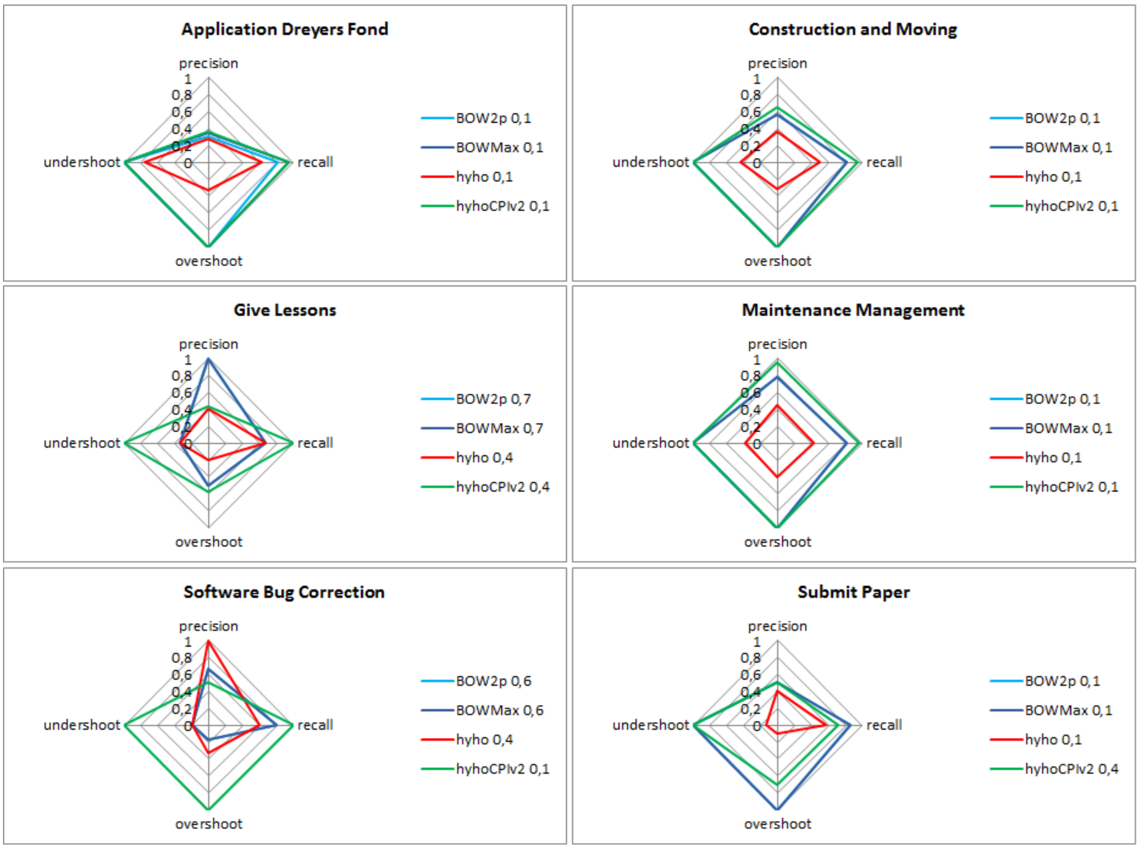
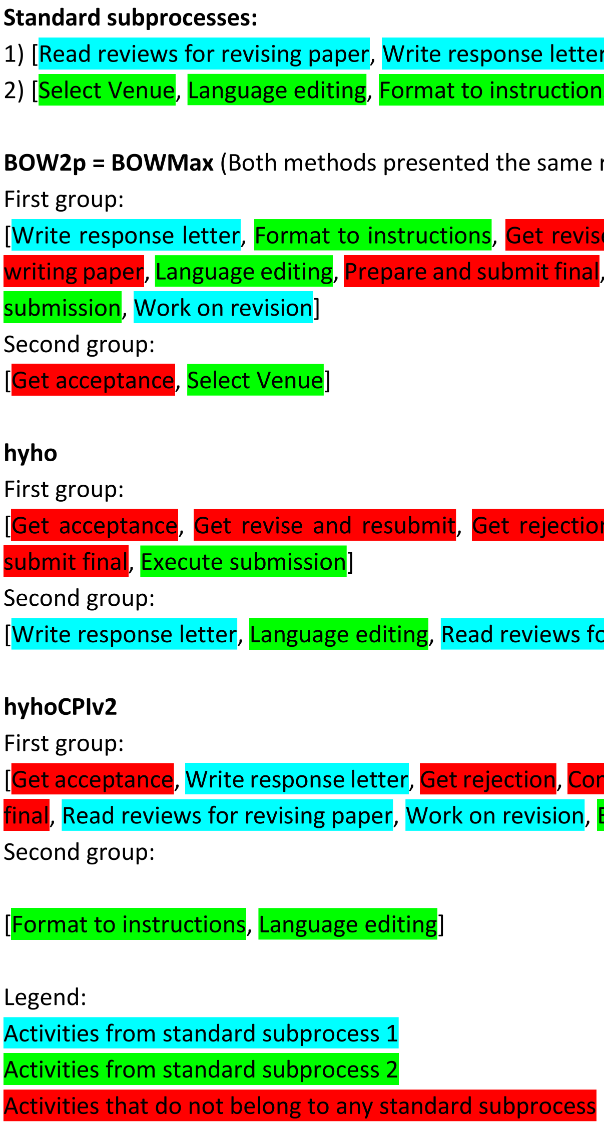


Figure 4.7: Method’s best results for each process model.

From Figure 4.7, it is possible to see that BOW2p, BOWMAx and hyhoCPIv2 reaches undershoot and overshoot values near the edges of the graph, while hyho tends to be near the center in most of the cases. Moreover, hyho presented lower precision and recall values in comparison to other methods. This confirms that BOW2p, BOWMAx and hyhoCPIv2 tends to create larger groups (high recall), but with some activities that do not belong to the standard group (high overshoot). They also produce groups that are not present on the standards (high undershoot). hyho, on the other hand, tends to suggest smaller groups (lower recall). It was not possible to define a pattern for the behavior of the precision metric due its different variations among the methods and process models. An example of this behavior can be seen in the process “Submit Paper” shown in Figure 4.8.



Standard subprocesses:

1) [Read reviews for revising paper, Write response letter, Work on revision]

2) [Select Venue, Language editing, Format to instructions, Execute submission]

BOW2p = BOWMax (Both methods presented the same results for this model)

First group:

[Write response letter, Format to instructions, Get revise and resubmit, Get rejection, Complete writing paper, Language editing, Prepare and submit final, Read reviews for revising paper, Execute submission, Work on revision]

Second group:

[Get acceptance, Select Venue]

hyho

First group:

[Get acceptance, Get revise and resubmit, Get rejection, Complete writing paper, Prepare and submit final, Execute submission]

Second group:

[Write response letter, Language editing, Read reviews for revising paper, Work on revision]

hyhoCPIv2

First group:

[Get acceptance, Write response letter, Get rejection, Complete writing paper, Prepare and submit final, Read reviews for revising paper, Work on revision, Execute submission]

Second group:

[Format to instructions, Language editing]

Legend:

Activities from standard subprocess 1

Activities from standard subprocess 2

Activities that do not belong to any standard subprocess

Figure 4.8: Groups comparison for the “Submit Paper” process model.

Table 4.4: Method’s best results for the “Submit Paper” process model.

process	method	precision	recall	f1	overshoot	undershoot
SubmitPaper	BOW2p	0,50	0,86	0,63	1,00	1,00
	hyho	0,40	0,57	0,47	0,40	0,57
	hyhoCPIv2	0,50	0,71	0,59	0,70	1,00

Figure 4.8 shows how the method hyho was penalized on precision and recall because it suggested a group that has five activities that did not belong to any reference group, although it produced a second group with all the activities from the reference group 1 with just one additional activity. The second group helps keeping low overshoot and undershoot.

Table 4.3 resumes the overall behavior of each method on the “Submit Paper” process model. Methods BOW2p and hyhoCPIv2 obtained reduced precision because they produced groups with too many activities (some of them were undesired), while recall was improved because these large groups also contained activities from the standard groups. Overshoot and undershoot where higher due the presence of superfluous and missing activities (marked as red in Figure 4.8) in its individual groups. The hyho method presented the lowest f-measure, and as seen previously obtained the lowest results for overshoot and undershoot.

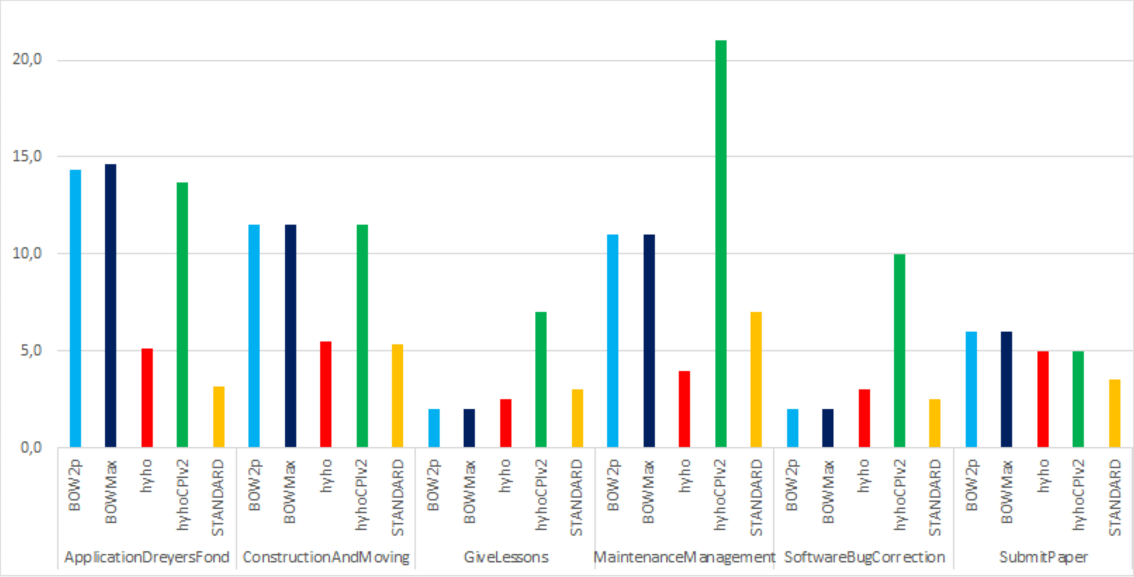


Figure 4.9: Average group size comparison.

Figure 4.9 compares the average of the group sizes, where it is possible to verify that the hyho method presented lower values in four out of the six process models. The other two cases are process models with few activities in standard groups: “Software Bug Correction” (2.5 activities per group in average) and “Give Lessons” (3.0 activities per group in average). hyho was the method with the most similar average group size when compared to the standard models.

In order to select a method for a case study considering all four subprocess metrics, BOWMax, BOW2p and hyhoCPIv2 were compared to hyho, which demonstrated a regular pattern with lower undershoot and overshoot values. Although hyho has not been the winner on precision and recall in all cases, it is interesting to know at least if its f-measure results are comparable to other methods when we limit these methods to the same values of undershoot and overshoot presented in the best results of the hyho method. In a nutshell, knowing that hyho is stable for undershoot and overshoot, the overall idea is to evaluate whether hyho presents an f-measure comparable to the remaining methods. If so, we consider hyho to be a suitable candidate method for our case study with real life data.

Figure 4.10 presents a diamond graph with the average values per method, considering the best f-measure for each method in the six selected process models, but limited by the undershoot and overshoot values from the best f-measure results from hyho method. It is worthy noticing that precision and recall from hyho are comparable (and even higher than, in some cases) to other methods.

From this analysis, the hyho method for the case study, since it returned groups with less missing or superfluous activities, with a precision and recall comparable to the other methods.

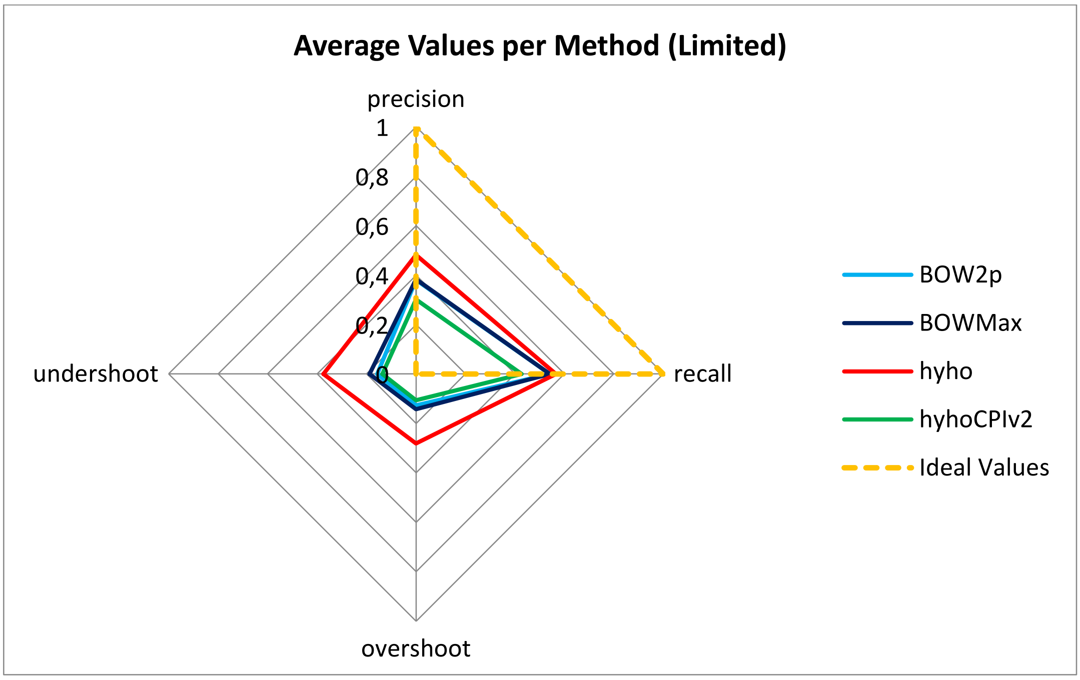


Figure 4.10: Average Results per method, limited by hyho's overshoot and undershoot.

4.6 Method application on Declarative Models visualization

Once defined the method for creating groups of activities, the second step is the creation of abstractions over a declarative model. The approach from (DEBOIS et al, 2014) that presented a technique for hierarchical declarative modeling for DCR Graphs, and the map metaphors were considered. The proposal is to add a more abstract representational layer over a low-level declarative map. It is intended to reduce diagram complexity presented to users, and to not harass their cognitive limits, which means the number of diagram elements that can be comprehended at a time, limited by working-memory capacity. When this is exceeded, a state of cognitive overload ensues and comprehension degrades rapidly (MILLER, 1956).

To create abstraction layers for visualization, it is proposed to connect complex activities (resulting from the previous grouping strategy) to other complex or single activities, by substituting all constraints between them by a single arc. To enforce aggregation behavior, the arc has variable thickness proportional to the number of constraints that were aggregated (Figure 4.11). From statechart semantics (HAREL et al., 1987) it is stated that if the actual conditions and/or the topology of the arrows are too complex connecting states and superstates (more abstract states), one can omit the details from the chart and use a simple incomplete form of Fig. 4.12c. Remembering that the user will have to supply the full details separately, and a computerized support system for statecharts would be able to show 4.12c but would enrich it to 4.12a or 4.12b upon request. This concept will be applied on declarative maps for creating a simpler more abstract view of a process, letting a software system provide detailing when needed.

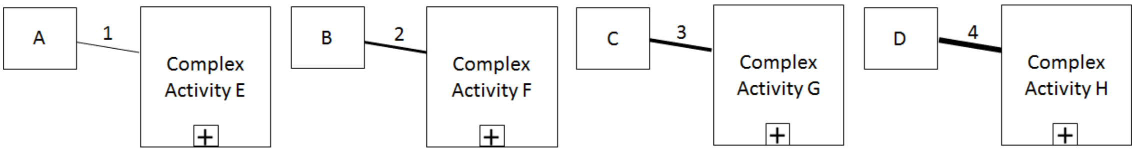


Figure 4.11: use of thicker arcs to denote more grouped constraints between activities.

The proposed representation is also in accordance with inter-group and intra-group classification of constraints (MAGGI et al, 2013). Intra-group refers to the class of constraints where the activities involved in a constraint all emanate from a single group, and inter-group refers to the class of constraints where the activities involved in a constraint belong to two different groups (Figure 4.13).

An important reason to create a process view as a layer over the low-level model instead of treating it as a subprocess embedded into the process model is the possibility of semantic losses caused by the hierarchy (ZUGAL et al, 2013). This occurs because some combinations of inter-group constraints cannot be represented by any other single constraint; also, different existence constraints over activities inside a group may be conflicting when trying to use just one constraint to represent them in their complex activity. For example, consider the situation when an activity A has an existence constraint an activity B has an absence constraint. In this situation, no single constraint template available can represent both behaviors. Other examples are presented in Figure 4.14: there is no single constraint that is equivalent to the combination of a *chain response* and a *not co-existen*ce constraints or a *response* and *not succession* from an activity to two distinct other activities that belongs to the same group.

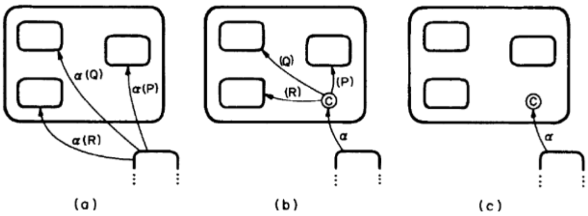


Figure 4.12: Statechart semantics for abstract states (HAREL et al., 1987).

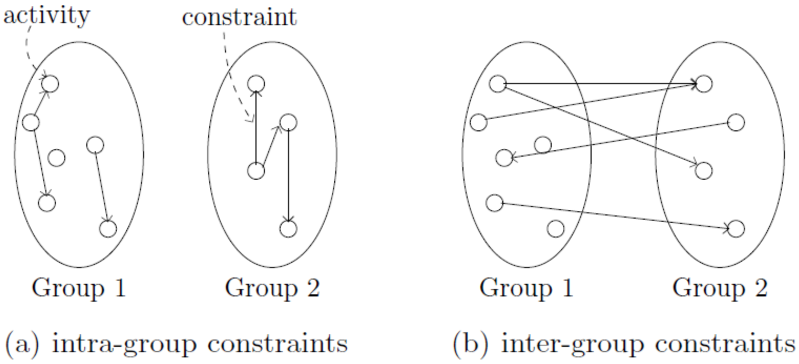


Figure 4.13: Inter- and intra-group constraints (MAGGI et al., 2013).

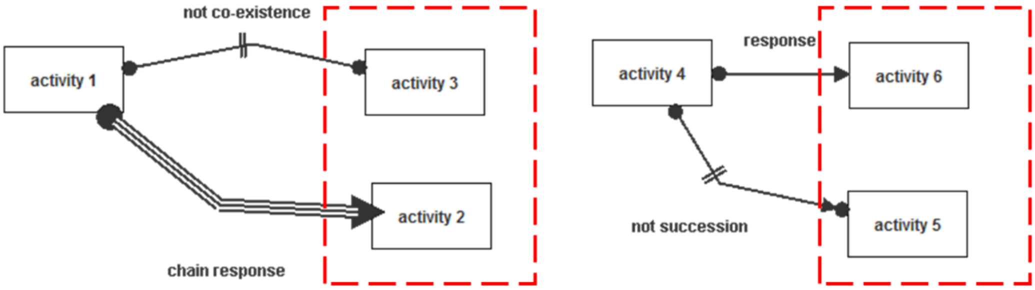


Figure 4.14: Activities that do not have a single equivalent aggregating constraint.

4.7 Method Execution Example on an Artificial Event Log

To show how the method works, this section presents an example process model, and all steps needed to generate the declarative map with abstraction, starting from pre-processing an event log. The main objective of this example is to observe how a declarative process model turns less complex after the use of abstractions suggested by the presented method. The declarative process model “How to prepare oneself and materials for teaching pupils” was chosen from literature (HAISJACKL et al., 2013) to perform this example. For this, there was created a single group of activities in order to focus on the illustration of the method application.

The process was modeled and simulated in CPNTools[16], generating 5,000 traces of an artificial event log that was used as input to DeclareMiner's plugin from ProM to discover a Declarative model. The plugin parameters were set to "Min. Support" = 50 and "alpha" = 50, which are the default parameters of DeclareMiner, and no additional filters were applied after the discovery. Figure 4.15 shows the discovered Declare Map.

After discovering the declarative model for the original event log, the grouping method was applied. First, each unique activity label within the event log was extracted and stored in a list, as in Figure 4.16, which shows a fragment, with three cases, from an example event log.

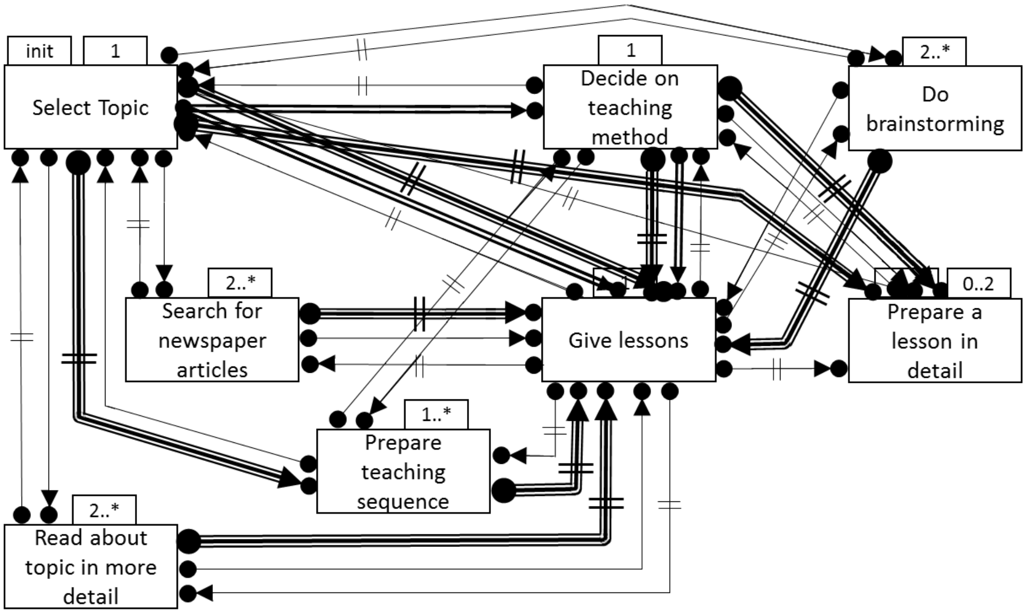
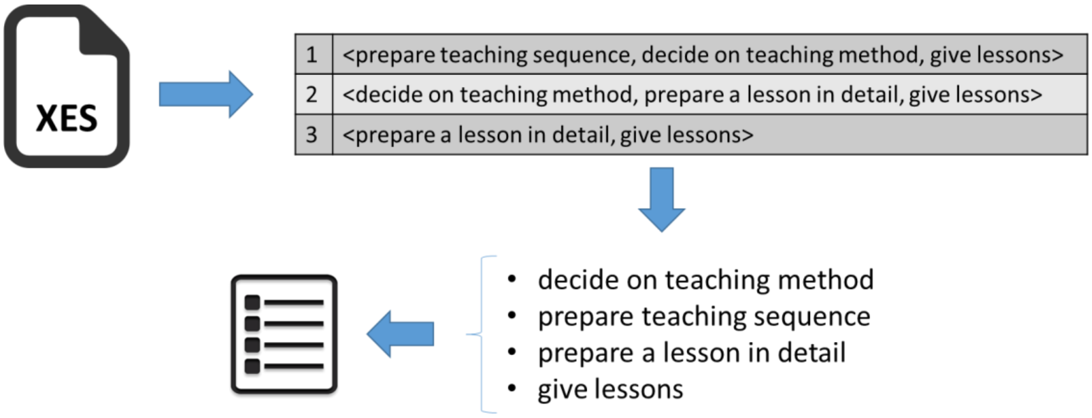

init
1
Select Topic
1
Decide on teaching method
2..*
Do brainstorming
2..*
Search for newspaper articles
Give lessons
0..2
Prepare a lesson in detail
1..*
Prepare teaching sequence
2..*
Read about topic in more detail

Figure 4.15: Discovered Declare map from the example event log.



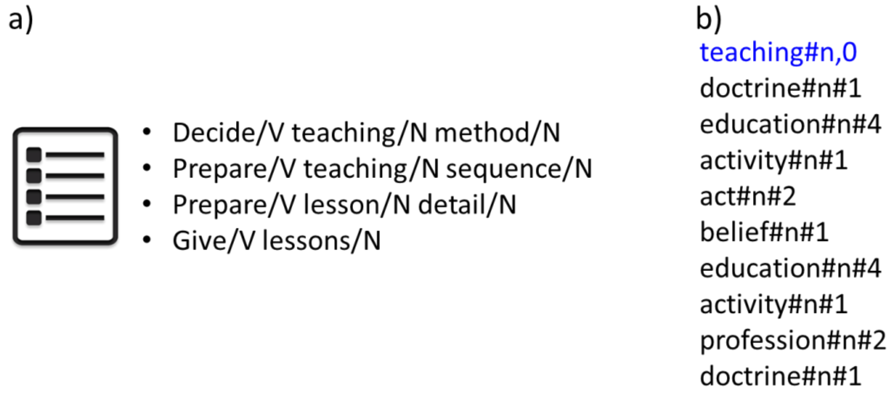
1	<prepare teaching sequence, decide on teaching method, give lessons>
2	<decide on teaching method, prepare a lesson in detail, give lessons>
3	<prepare a lesson in detail, give lessons>

- decide on teaching method
- prepare teaching sequence
- prepare a lesson in detail
- give lessons

Figure 4.16: Obtaining the activities list from a process event log.

[16] The tool is available at http://cpntools.org/.

Each label was then annotated with its respective part-of-speech tag. Hypernyms and holonyms were located for each word (noun or verb) from the list (Figure 4.17a). Figure 4.17b shows a list of WordNet hypernyms and holonyms synsets for the “teaching” word. A distance limit of 2 was set from the origin word. The resulting hypernyms and holonyms found were stored for each word in the label list.

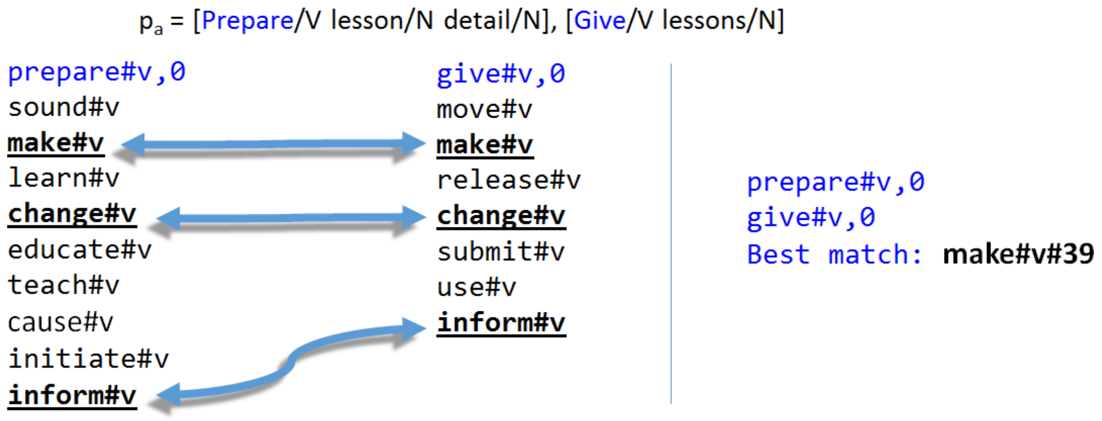


- Decide/V teaching/N method/N
- Prepare/V teaching/N sequence/N
- Prepare/V lesson/N detail/N
- Give/V lessons/N

```
teaching#n,0
doctrine#n#1
education#n#4
activity#n#1
act#n#2
belief#n#1
education#n#4
activity#n#1
profession#n#2
doctrine#n#1
```

Figure 4.17: Annotated labels (a) and hypernyms and holonyms for the “teaching” word (b).

A combination of pairs of all activity labels is then generated. For each activity pair, all noun-to-noun or verb-to-verb word pairs were considered and a match of their common hypernyms or holonyms was performed. Note that more than one concept may match. In order to select only the best match between the words, Word Sense Disambiguation was applied to verify which one is more appropriated in the context of the business process. The best match can be obtained (Figure 4.18) by directly running the algorithm WordNet::SenseRelate::WordToSet (MICHELIZZI et al., 2005) taking as input parameters the set of common hypernyms/holonyms (*make*, *change*, *inform*) and the process context consisting in a set of all words from the entire activity label’s list.



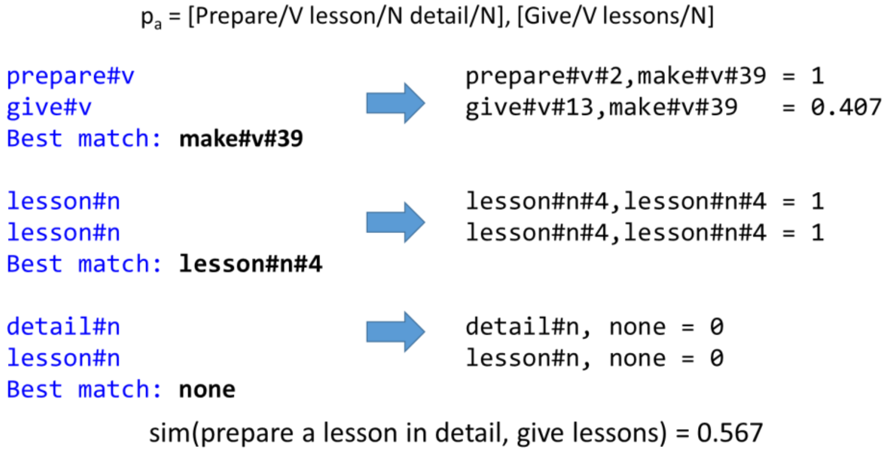
p_a = [Prepare/V lesson/N detail/N], [Give/V lessons/N]

```
prepare#v,0      give#v,0
sound#v          move#v
make#v           make#v
learn#v          release#v
change#v         change#v
educate#v        submit#v
teach#v          use#v
cause#v          inform#v
initiate#v
inform#v
```

```
prepare#v,0
give#v,0
Best match: make#v#39
```

Figure 4.18: Hypernyms and holonyms matching for word pairs in an activity label pair.

Afterwards, the semantic similarity from a word and its best match is calculated using Lin's similarity metric. For calculating the similarity between a pair of activity labels, Equation 7 was applied.



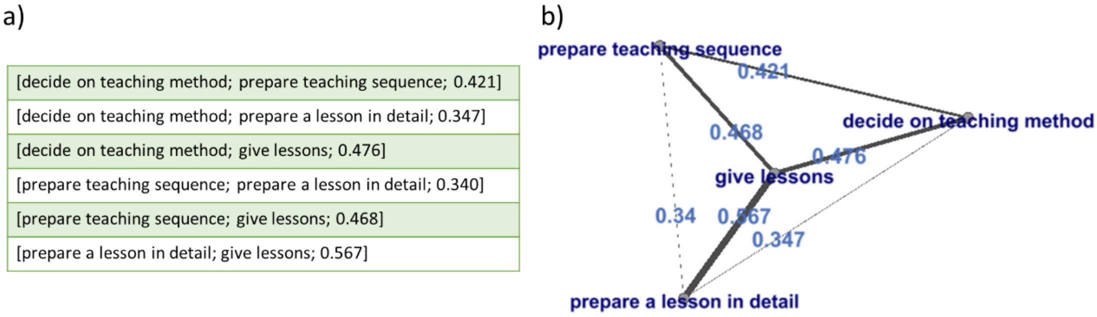
p_a = [Prepare/V lesson/N detail/N], [Give/V lessons/N]

```
prepare#v                            prepare#v#2,make#v#39 = 1
give#v                               give#v#13,make#v#39   = 0.407
Best match: make#v#39

lesson#n                             lesson#n#4,lesson#n#4 = 1
lesson#n                             lesson#n#4,lesson#n#4 = 1
Best match: lesson#n#4

detail#n                             detail#n, none = 0
lesson#n                             lesson#n, none = 0
Best match: none
```

sim(prepare a lesson in detail, give lessons) = 0.567

Figure 4.19: Similarity calculus for an activity labels pair.

After calculating the similarity for all words, the average semantic relatedness value for the activity pair is calculated (Figure 4.19). The activity pair and its average semantic relatedness value are stored in a set. With this set (Figure 4.20a), an undirected weighted graph is built to help define the groups of similar activities (Figure 4.20b). This is done by analyzing all the possible fully connected subgraphs, or cliques, and summing the edges of each clique (Figure 4.21).



a)

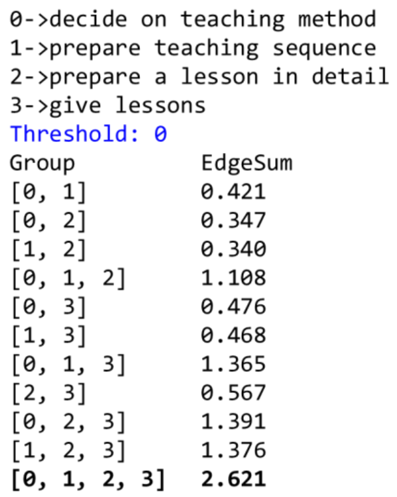
[decide on teaching method; prepare teaching sequence; 0.421]
[decide on teaching method; prepare a lesson in detail; 0.347]
[decide on teaching method; give lessons; 0.476]
[prepare teaching sequence; prepare a lesson in detail; 0.340]
[prepare teaching sequence; give lessons; 0.468]
[prepare a lesson in detail; give lessons; 0.567]

b)

prepare teaching sequence
0.421
0.468
decide on teaching method
0.476
give lessons
0.34
0.567
0.347
prepare a lesson in detail

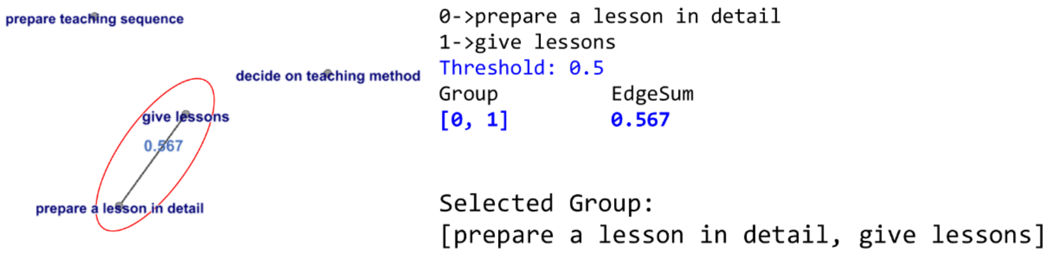
Figure 4.20: Activity pairs and their similarity values, using a (a) textual representation and a (b) graph representation.

Iteratively, the subgraph with the highest value is stored in a list and removed from the original graph, until there are no more edges on the graph. Each subgraph stored in the list is a complex activity candidate (Figure 4.22). For this example, it finished with a single group with two activities ("Give lessons" and "Prepare a lesson in detail"), due the user decision to use a minimum similarity value of 0.5 between activities.



```
0->decide on teaching method
1->prepare teaching sequence
2->prepare a lesson in detail
3->give lessons
Threshold: 0
Group          EdgeSum
[0, 1]         0.421
[0, 2]         0.347
[1, 2]         0.340
[0, 1, 2]      1.108
[0, 3]         0.476
[1, 3]         0.468
[0, 1, 3]      1.365
[2, 3]         0.567
[0, 2, 3]      1.391
[1, 2, 3]      1.376
[0, 1, 2, 3]   2.621
```

Figure 4.21: Edge sum for each clique on the graph.


prepare teaching sequence
decide on teaching method
give lessons
0.567
prepare a lesson in detail

```
0->prepare a lesson in detail
1->give lessons
Threshold: 0.5
Group          EdgeSum
[0, 1]         0.567


Selected Group:
[prepare a lesson in detail, give lessons]
```

Figure 4.22: Selected group using a minimum similarity value of 0.5.

With the previously discovered declarative model, the abstraction method was applied based on the group discovered. We substitute “Give lessons” and “Prepare a lesson in detail” activities by a new complex activity that was manually labeled as “Prepare and Give lessons”. In addition, all constraints that connected these two activities to other activities of the models were removed by the aggregating arcs with variable thickness (for example, the constrains *not succession(prepare teaching sequence, give lessons)* and *not chain succession(give lessons, prepare teaching sequence)* were encapsulated into an unique arc between *prepare teaching sequence* and *prepare and give lessons*). Figure 4.23 presents the resulting Declare map with abstractions. Numbers over the arcs represent the relation constraint quantities that were abstracted.

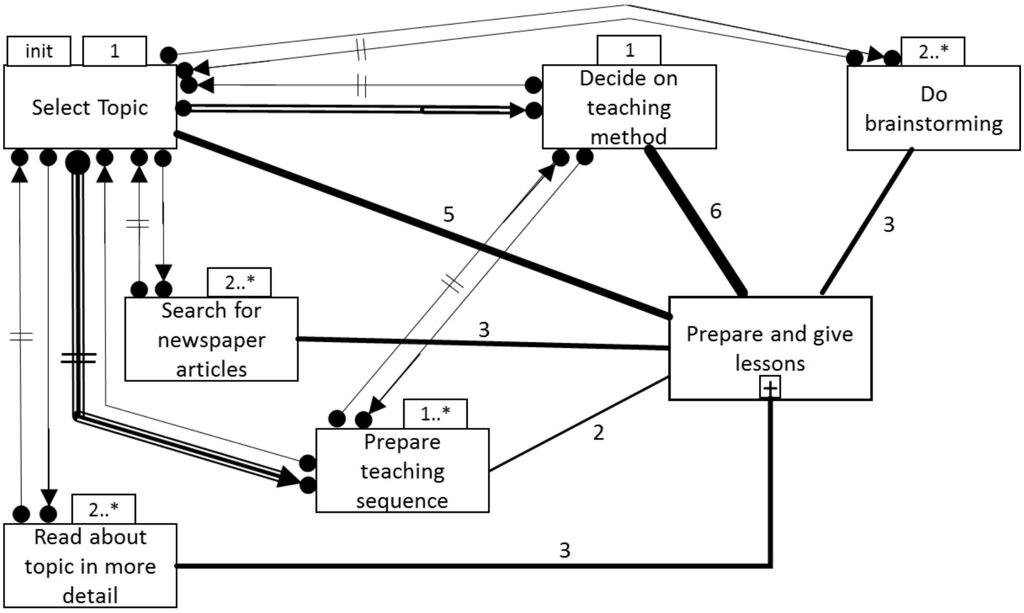

init
1
Select Topic
1
Decide on teaching method
2..*
Do brainstorming
5
6
3
2..*
Search for newspaper articles
3
Prepare and give lessons
1..*
Prepare teaching sequence
2
2..*
Read about topic in more detail
3

Figure 4.23: Declare map with abstractions.

In a visual inspection, a process analyst can see that the model with abstractions is more readable, due the lower quantity of activities and constraints it presents in comparison to the model without abstractions. It is possible to view the inter-group constraints by selecting an aggregating arc (Figure 4.24). Intra-group constraints within a complex activity are visualized by selecting a complex activity (Figure 4.25).

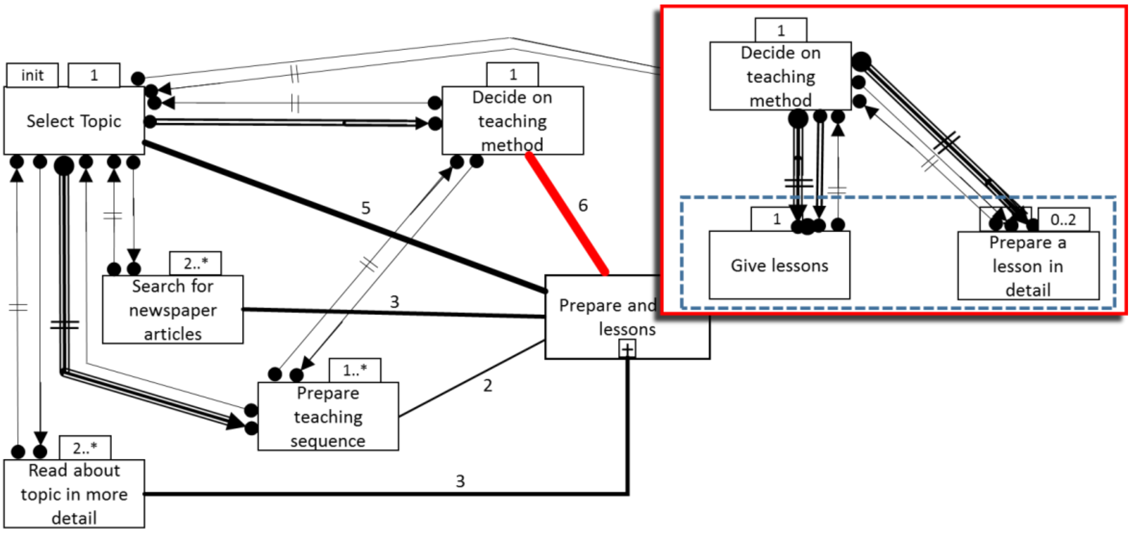

init
1
Select Topic
1
Decide on teaching method
5
6
2..*
Search for newspaper articles
3
Prepare and lessons
1..*
Prepare teaching sequence
2
2..*
Read about topic in more detail
3
1
Decide on teaching method
1
Give lessons
0..2
Prepare a lesson in detail

Figure 4.24: Details of inter-group constraints between activity “Decide on teaching method” and complex activity “Prepare and give lessons”.

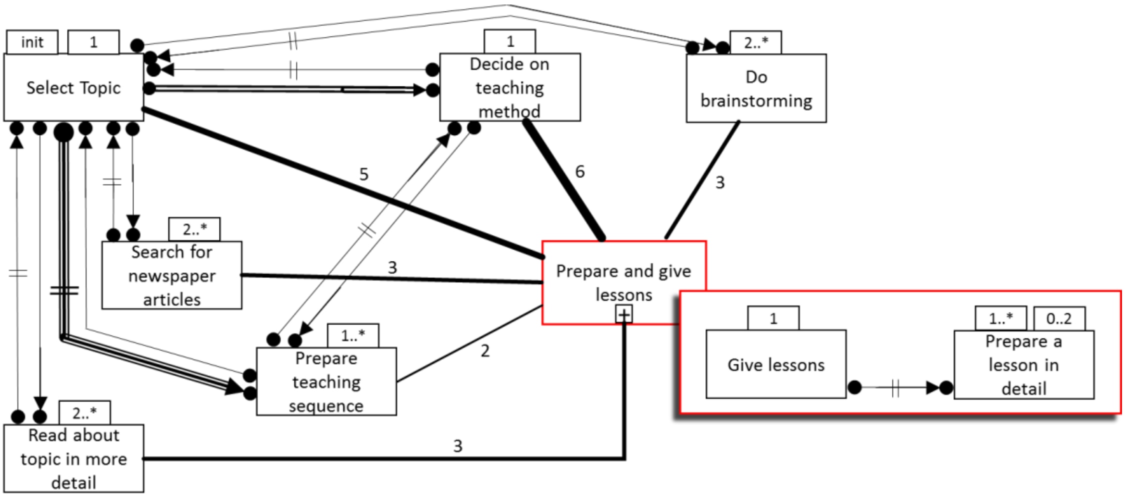

init
1
Select Topic
1
Decide on teaching method
2..*
Do brainstorming
5
6
3
2..*
Search for newspaper articles
3
Prepare and give lessons
1..*
Prepare teaching sequence
2
1
Give lessons
1..*
0..2
Prepare a lesson in detail
2..*
Read about topic in more detail
3

Figure 4.25: Details of intra-group constraints within “Prepare and give lessons” complex activity.

4.8 Final Remarks

The manual creation of groups may consider other aspects rather than semantics, e.g., the sequence flow, process data and user roles, and also tacit knowledge. As the processes are declarative, the structural approach is not recommended, since it is based on control flow sequences (Zugal et al., 2013). In addition, the proposed approach assumes that process data and user information are not available. This way, this proposal creates groups by looking just for the activity labels, in a minimum information scenario that sometimes is present on real life situations. The proposed approach helps users by producing simpler models with abstractions, where each group relates activities by their semantics. If additional process information is available, such as temporal aspects, user roles and process data, it may be possible to extend the method to also consider these aspects in order to improve the suggestion of groups of activities.

The natural language processing phase may introduce some errors, such as pos-mistagging a verb as a noun and vice-versa. In addition, not always an activity name is written as a complete sentence, what make this phase more challenging. In scenarios where local context is poor or vague, word sense disambiguation may also lead to incorrect suggestions for the best common hypernymy or holonymy. All these problems depend on the labeling quality, which should be an important concern for business analysts when they design process models (LEOPOLD, 2013).

For the semantic similarity measures between activity labels, the labeling quality is a major factor of success. When working with process models with poor labeling, e.g., single wording, absence of an action or a business object, it is expected that the results will decay. To avoid this situation, it is recommended to follow process modeling guidelines (LEOPOLD, 2013) to improve the quality and consistency of process models.

The findings show that the minimum value for the semantic similarity between activities of the best groups falls in the range [0.1, 0.7], considering the six analyzed models. It is possible to say that the similarity values are not higher because this would mean that activities are nearly equal, what is expected to not be frequent within a single model. Regarding the subprocess metrics applied on the analysis, instead of using the diamond graph analysis, it will be interesting to develop a single metric capable of gathering all aspects that influence on the quality of subprocesses, i.e., the precision, recall, undershoot and overshoot. Similar to f-measure, which summarizes precision and recall, a new subprocess metric should also consider the effects of undershoot and overshoot in a single measure, what could turn the analysis of automatically generated subprocesses easier.

About the group formation, manual grouping usually considers more information beyond the semantics, even the personal experience and judgment of the modeler counts when he or she decides which activities should be grouped. However, this proposed approach intends to help domain specialists or beginner practitioners to have a better understanding on declarative process models, suggesting possible groups that make the models less complex and easy to understand. When there is no previous knowledge available about the model (a situation that may happen on process mining scenarios), this approach produces interesting views about the process behavior that may be useful for analysis and further improvement of business processes.

Chapter 5 - Case Study

This chapter explains the results obtained by the application of the proposed method on a real life dataset. The main goal is to create groups of activities that can be used to build an abstraction layer over an automatically discovered declarative map.

5.1 Case Study Setup

The case study was planned to execute and evaluate the steps of the proposed method in a real life setting. It consists in executing the proposed algorithms to create groups of activities using an event log from the execution of a real world process. The execution was followed by a two-phase evaluation. The first phase conducted a domain expert's evaluation of the usefulness of the groups suggested by the activity grouping technique. After discovering a declarative map using the same event log and applying the visualization techniques from Section 4.6 to create an abstraction layer, the second phase of the evaluation compares the original discovered declare map against its representation with abstractions, by using complexity metrics. This last evaluation aims to quantitatively assess if the solution proposal is in accordance with the research hypothesis. It is *"IF semantics is used to find abstraction and aggregation relations between activity labels in a declarative model THEN it will be possible to generate groups of activity labels, with useful meanings for domain stakeholders, enabling the creation of abstractions over declarative process models in order to produce process views with reduced complexity"*.

5.1.1 Event log characterization

Declarative models are particularly effective for some non-conventional kinds of processes, which involves the production of valuable but intangible products, such as knowledge (DI CICCIO et al, 2013). This type of processes is flexible, dynamic and subject to change, and frequently referred to as knowledge-intensive processes, or KIPs (DAVENPORT, 2005).

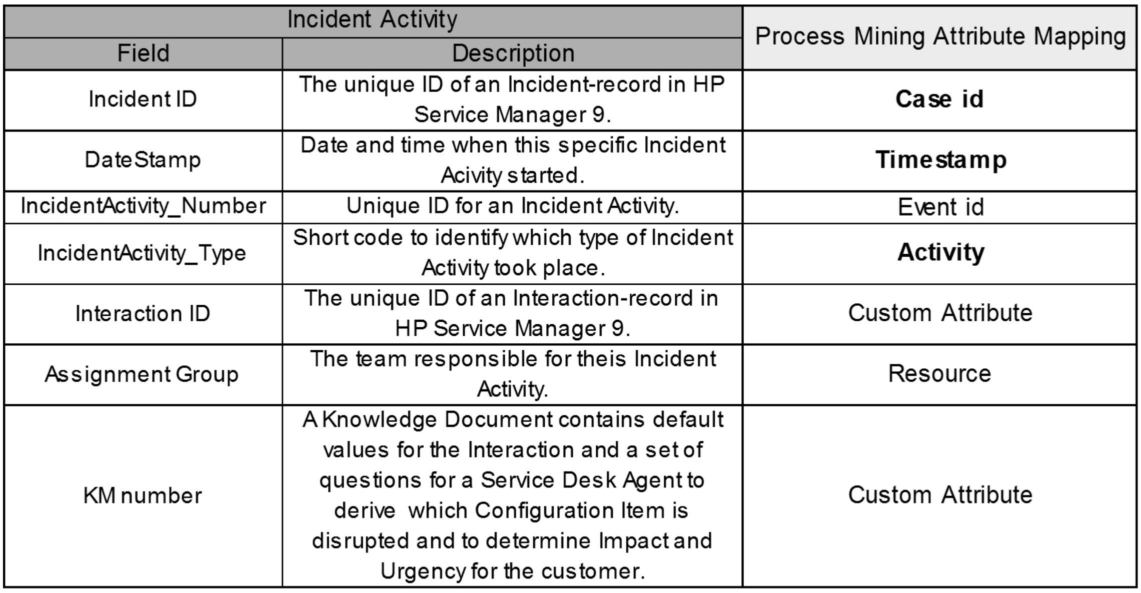
In order to evaluate the previous proposed method in a real life environment, an event log with the aforementioned characteristics was selected from the Business Process

Intelligence Challenge 2014 description (VAN DONGEN, 2014). This data set contains information about the execution of processes that implements ITIL in a Dutch bank, from where it was selected a particular file containing records related to Information and Communications Technology (ICT) incident activities' lifecycle.

From the data model of this process, it is possible to observe the minimum attributes required for process mining (VAN DER AALST, 2011). Table 5.1 presents this mapping, with the minimum required attributes written in bold font.

The "Incident Activity" event log contains information about the ICT incidents' lifecycle: from opening an incident after user interaction, passing through diverse possible actions to solve the problem depending on the nature of the incident, until it is closed.

Table 5.1: Event log data model's mapping to process mining attributes.



Incident Activity		Process Mining Attribute Mapping
Field	Description	
Incident ID	The unique ID of an Incident-record in HP Service Manager 9.	**Case id**
DateStamp	Date and time when this specific Incident Acivity started.	**Timestamp**
IncidentActivity_Number	Unique ID for an Incident Activity.	Event id
IncidentActivity_Type	Short code to identify which type of Incident Activity took place.	**Activity**
Interaction ID	The unique ID of an Interaction-record in HP Service Manager 9.	Custom Attribute
Assignment Group	The team responsible for theis Incident Activity.	Resource
KM number	A Knowledge Document contains default values for the Interaction and a set of questions for a Service Desk Agent to derive which Configuration Item is disrupted and to determine Impact and Urgency for the customer.	Custom Attribute

Disco[17] tool was used to collect some event log statistics, by importing the comma-separated text file with incident activities' data. Table 5.2 shows the overall information about this event log. There are 46.606 cases, and 21.756 different case variants, what points to a process model with high variability, indeed an interesting candidate for Declarative modeling. In order to confirm the inappropriateness of an imperative approach for this type of process, a process Map was generated by Disco, showing all activities and only 30% of the paths. Figure 5.1 shows this imperative map, showing how difficult it is to understand the relations among the activities, due the high quantity of paths. Disco was also used to convert the comma-separated text file to the XES format to be used on process mining.

[17] For Disco tool refer to http://www.fluxicon.com.

Table 5.2: Overall statistics from Incident Activity event log.

Information	Value
Events	466.665
Cases	46.606
Activities	39
Case Variants	21.756

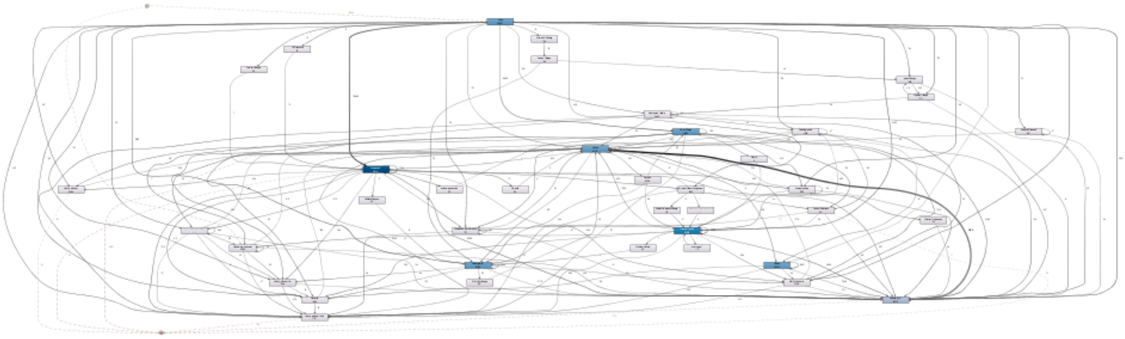


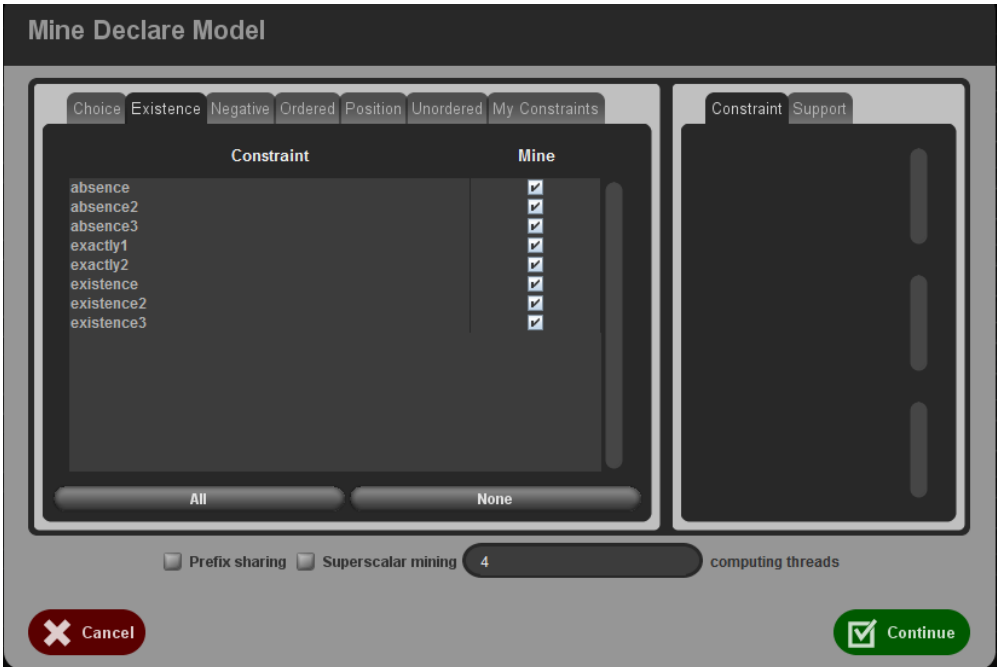
Figure 5.1: Imperative Map generated by Disco from the BPIC 2014 ICT incident activities' lifecycle dataset, showing all activities and just 30% of the paths.

5.1.2 Process Mining Configuration and Running

For generating a declarative model, Declare language was chosen because it is the only which have available mining algorithms in the literature. Since this event log has more than 46.000 cases and ProM's DeclareMiner plugin is known as the slowest available (WESTERGAARD et al., 2013), it was discarded. MINERFul++ (DI CICCIO et al., 2013) cannot tackle complex constraints. Therefore, UnconstrainedMiner was chosen for process discovery. It offers both good performance and the possibility to discover all Declare constraint templates. Figure 5.2 shows an example screen of the UnconstrainedMiner plugin in ProM's environment.

For the first time running, all Declare constraint templates where selected, and four computing threads were set to improve speed. All these steps were executed in an Intel Core i5 64 Bits CPU, with 2GB RAM computer.

Unconstrained Miner discovers all possible constraints from an event log, resulting in a large list of constraints that needs to be further filtered. A fragment from its output is shown in Table 5.3. From this table, it is possible to see the constraint templates and its parameters, the positive, negative and dependent support, from where it is possible to calculate the frequency based support and confidence of each constraint.



Constraint	Mine
absence	☑
absence2	☑
absence3	☑
exactly1	☑
exactly2	☑
existence	☑
existence2	☑
existence3	☑

☐ Prefix sharing ☐ Superscalar mining 4 computing threads

Figure 5.2: UnconstrainedMiner plugin example screen.

Table 5.3: Example output from UnconstrainedMiner.

constraint	parameters	positive	negative	dependent	confidence	support
alternate precedence	[[Assignment], [Caused By CI]]	34159	0	27079	0.7927	0.581
not chain succession	[[Assignment], [Caused By CI]]	27508	0	18110	0.6584	0.3886
succession	[[Assignment], [Caused By CI]]	45319	0	26152	0.5771	0.5611
not chain succession	[[Assignment], [Closed]]	38438	0	35670	0.928	0.7654
alternate succession	[[Assignment], [Closed]]	46387	0	23845	0.514	0.5116

Thus, after the discovery, it is necessary to filter the plugin's output in order to obtain meaningful information. In addition, there is no single correct model on such process mining initiatives where stakeholders expect that valuable insights will emerge by analyzing event data. The adequacy of a discovered model will depend on the parameters used and on the judgment of the process analyst (VAN DER AALST, 2011). Thus, this step consisted on tuning a set of parameters to obtain a Declare map with the goal to represent the most frequent and confident constraints. We have tested scenarios with a constraint support from 10% to 50%, constraint confidence from 10% to 80%, and activity frequency from 0% to 100% and selected a set of parameters that fulfills this goal. The selected parameters are: constraint support higher than 20%, constraint confidence higher than 50% and activity frequency higher than 1%. This setting can be justified due the high number of different cases, where few activities have a high frequency rate (Table 5.4), implying on their constraints to also have limited support and confidence due the high size of the event log. When testing with higher parameters, the number of discovered activities and constraints was severely reduced, implying on not capturing potential important behavior.

Due to the size of the event log, a high quantity of choice and exclusive choice constraints were discovered that are not useful in practice, such as choice (Open, Closed), which means that the user need to choose between opening or closing an incident that are mandatory activities with frequency above 99% over the cases. So, these two types of constraints were not considered on the resulting Declare Map. Due to the inability of Unconstrained Miner to consider known techniques to improve Declare Maps, the resulting Declare model was post processed using two improvement techniques (MAGGI et al., 2013): weaker constraints removal and transitive reduction. From 105 constraints from the raw discovery, the post processing ended with 59 constraints. This way, it was possible to generate a simpler model in comparison to the raw set of constraints discovered by UnconstrainedMiner. The Declare Map without abstractions is presented in the results (Section 5.2.2), where it was compared to another Declare Map with abstractions built through the algorithms proposed in Chapter 4.

5.2 Proposed Method Execution

The activity labels from the "Incident Activity" event log were extracted and stored in a plain text file. With this activity list, Algorithm 1 (configured with hyho similarity) calculated the semantic similarity between the selected activity labels. The output was a set of pairs of activities with their respective semantic similarity value, which were used to run Algorithm 2, which in turn, generated groups' suggestions. The minimum semantic similarity value to limit group generation was set to 0.3. For the selection of this parameter, there were tested different values, ranging from 0.1 to 1.0, in steps of 0.1. As the choice for a particular set of groups is a user decision, and as a single group set was needed to perform the experiment, the experiment designer made a choice for the 0.3 value, which should be evaluated by domain specialists, in order to confirm its usefulness. As a partial step, Figure 5.3 shows the resulting graph used to find cliques and then propose groups. Due the high quantity of edges on the graph, it was difficult to manually find and sum the edges of each clique. Algorithm 2 helped automating this task. Then, the following groups were proposed:

- Group 1: Vendor Reference, Vendor Reference Change, Communication with vendor, Pending vendor;

- Group 2: Reassignment, Assignment;
- Group 3: Mail to Customer, Communication with customer;
- Group 4: Quality Indicator Fixed, Quality Indicator, Quality Indicator Set;
- Group 5: Service Change, Status Change, Contact Change, Notify By Change, Impact Change.

Since only activities with frequency higher than one percent were considered, only 25 (out of 39) activities from the entire event log appeared in the produced Declarative Model. Table 5.4 shows the frequency distribution of each selected activity.

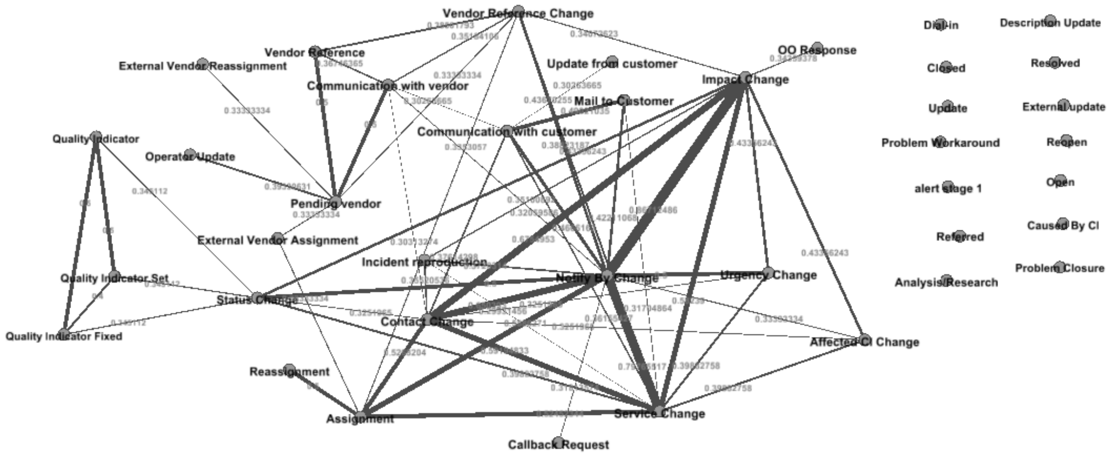


Figure 5.3: Resulting graph using a 0.3 minimum semantic similarity value between activities.

5.2.1 Evaluation of the suggested groups by Specialists

Since the groups are a result of a user decision on defining a minimum semantic similarity value, it was important to evaluate if the suggested groups are really meaningful. Therefore, an online survey[18] was developed. The survey was targeted to domain specialists, in order to gather their opinion about the suggested groups . This way, professionals with experience on Information Technology Service Management (ITSM) were asked to participate on this evaluation by answering the questions proposed, as detailed in the next Section.

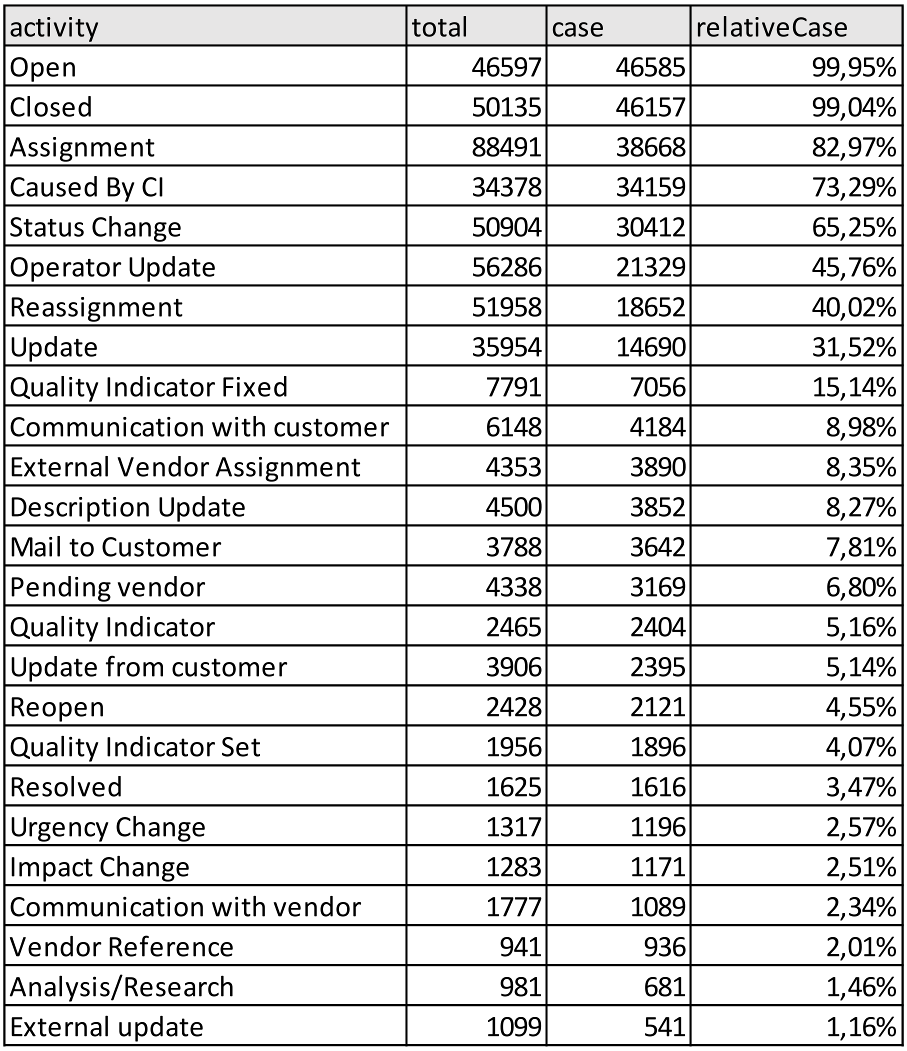
5.2.1.1 Survey design

The design of the online survey starts with a welcome page with information about its main goal, followed by instructions for completing the evaluation, a contextualization of the problem and the dataset used, and also information about the evaluation criteria of

[18] The evaluation survey is available at: http://goo.gl/vAv56U.

the suggested groups. These criteria are: a) if a suggested group contains activities with a common goal, b) if there is any activity to be included in a group that was not automatically included and c) if there is any activity belonging to a group that should be removed as it is not related to other activities in the same group. In addition, a qualification form is presented to verify the user experience on BPM, ITIL and ITSM.

Table 5.4: List of activities with case frequency (“relativeCase” field) above 1%.

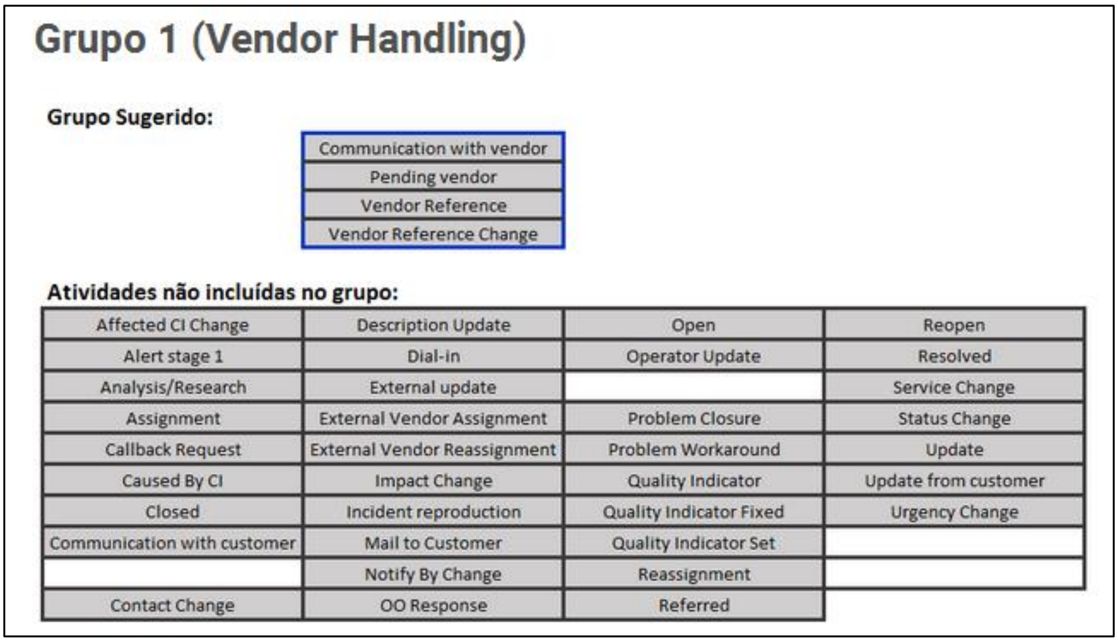


activity	total	case	relativeCase
Open	46597	46585	99,95%
Closed	50135	46157	99,04%
Assignment	88491	38668	82,97%
Caused By CI	34378	34159	73,29%
Status Change	50904	30412	65,25%
Operator Update	56286	21329	45,76%
Reassignment	51958	18652	40,02%
Update	35954	14690	31,52%
Quality Indicator Fixed	7791	7056	15,14%
Communication with customer	6148	4184	8,98%
External Vendor Assignment	4353	3890	8,35%
Description Update	4500	3852	8,27%
Mail to Customer	3788	3642	7,81%
Pending vendor	4338	3169	6,80%
Quality Indicator	2465	2404	5,16%
Update from customer	3906	2395	5,14%
Reopen	2428	2121	4,55%
Quality Indicator Set	1956	1896	4,07%
Resolved	1625	1616	3,47%
Urgency Change	1317	1196	2,57%
Impact Change	1283	1171	2,51%
Communication with vendor	1777	1089	2,34%
Vendor Reference	941	936	2,01%
Analysis/Research	981	681	1,46%
External update	1099	541	1,16%

The evaluation consists on presenting all 39 activities from the Incident Activity dataset to the user, and highlighting a group suggested by the proposed method. Figure 5.4 shows an example of a group presentation for the specialist’s evaluation. At this point, the participant is asked to confirm if the highlighted activities have a common objective and if they can be nested in a group. In addition, the user may add missing activities or remove undesired activities from the group. This step is repeated for all five suggested groups for this dataset.

Also, some manually created groups with the remaining activities were introduced to verify if the users are able to perceive any difference between the automated suggestions and manually created ones.

After evaluating the groups, the user is asked to inform if he or she would suggest any different group that was not previously presented. Finally, the users were asked to point which activity labels presented might have produced any comprehension problem. This last question is important to verify to what extent the labeling quality may impact the analysis of the groups.



Grupo 1 (Vendor Handling)

Grupo Sugerido:

- Communication with vendor
- Pending vendor
- Vendor Reference
- Vendor Reference Change

Atividades não incluídas no grupo:

Affected CI Change	Description Update	Open	Reopen
Alert stage 1	Dial-in	Operator Update	Resolved
Analysis/Research	External update		Service Change
Assignment	External Vendor Assignment	Problem Closure	Status Change
Callback Request	External Vendor Reassignment	Problem Workaround	Update
Caused By CI	Impact Change	Quality Indicator	Update from customer
Closed	Incident reproduction	Quality Indicator Fixed	Urgency Change
Communication with customer	Mail to Customer	Quality Indicator Set	
	Notify By Change	Reassignment	
Contact Change	OO Response	Referred	

Figure 5.4: Example of a group to be evaluated by a specialist.

Once the survey was closed, it was analyzed how much agreement among the respondents was achieved for each activity presented in the suggested groups.

5.2.1.2 Participant's Profile

The survey was answered by eight professionals. Figure 5.5 presents user's profile on interest topics for this evaluation. Y-axis represents the number of participants and x-axis the experience level. The experience level starts from 1 (lowest) to 5 (highest). Regarding BPM (a), all users reported medium to high experience on the topic (values among 3 to 5). For ITSM (b) just one user reported a low-to-medium experience, although the remaining users fell in between 4-5 score. For ITIL, that is a complimentary topic related to ITSM, a single user reported low-to-medium experience, another with medium experience (3-score) and the six remaining participants pointed a medium-to-high (4-score) experience. As these scores are self-evaluations from the users, they are subjected to personal judgments. As there is not a large quantity of users with low scores, we did not prune the answers for the user reporting low-to-regular (2-score) knowledge on ITSM and ITIL.

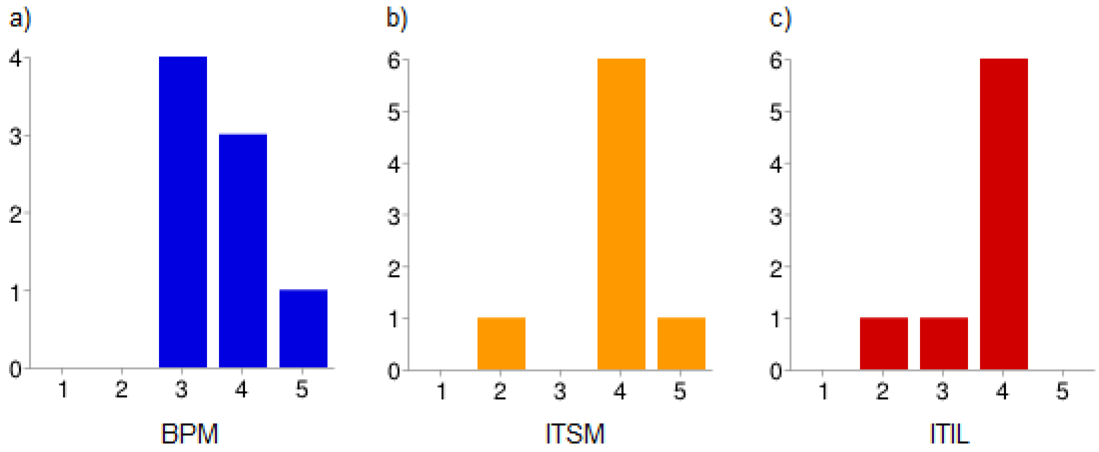


Figure 5.5: User's profile on interest topics for the survey.

5.2.1.3 Survey Results

For each of the five suggested groups, the survey participants were asked to confirm if the activity suggestions are meaningful to the domain. In addition, they were motivated to remove undesired activities that were proposed or including new activities in a group (from the 39 activities of the dataset) that were not considered by the automatic method. Finally, all results were consolidated for each group, and the frequency of each user confirmation was calculated in order to assess how many times the proposed activities were selected by the survey participants. Table 5.5 shows the results for each automatically proposed groups. For improved readability, activities that were not selected and do not belong to any group were omitted from the tables presented in Table 5.5. Activities highlighted in green are the automatic suggestions. All other activities are selections made by the participants. The "Hits" column points to the number of times an activity was selected and the "%" is the relative frequency of the selection considering eight evaluations.

Considering the five proposed groups, all suggested activities (green highlighted) presented a relative selection frequency above 88%, except for the "Vendor Reference Change" activity on Group 1, which was agreed by six users (75%). In addition, for the activities included by the participants, no more than two participants agreed on the inclusion, except for the "Update from customer" activity on Group 3 that was added by three users.

Table 5.5: User’s activities selection on automatically proposed groups.

a)

Group 1		
Activity	**Hits**	**%**
alert stage 1	1	13%
Communication with vendor	7	88%
Dial-in	1	13%
External Vendor Assignment	2	25%
External Vendor Reassignment	2	25%
Pending vendor	7	88%
Vendor Reference	7	88%
Vendor Reference Change	6	75%

b)

Group 2		
Activity	**Hits**	**%**
Assignment	8	100%
Closed	1	13%
External Vendor Assignment	2	25%
External Vendor Reassignment	2	25%
Open	1	13%
Reassignment	8	100%
Reopen	1	13%
Status Change	1	13%

c)

Group 3		
Activity	**Hits**	**%**
Callback Request	1	13%
Closed	1	13%
Communication with customer	8	100%
Mail to Customer	8	100%
Update from customer	3	38%

d)

Group 4		
Activity	**Hits**	**%**
Quality Indicator	8	100%
Quality Indicator Fixed	8	100%
Quality Indicator Set	8	100%

e)

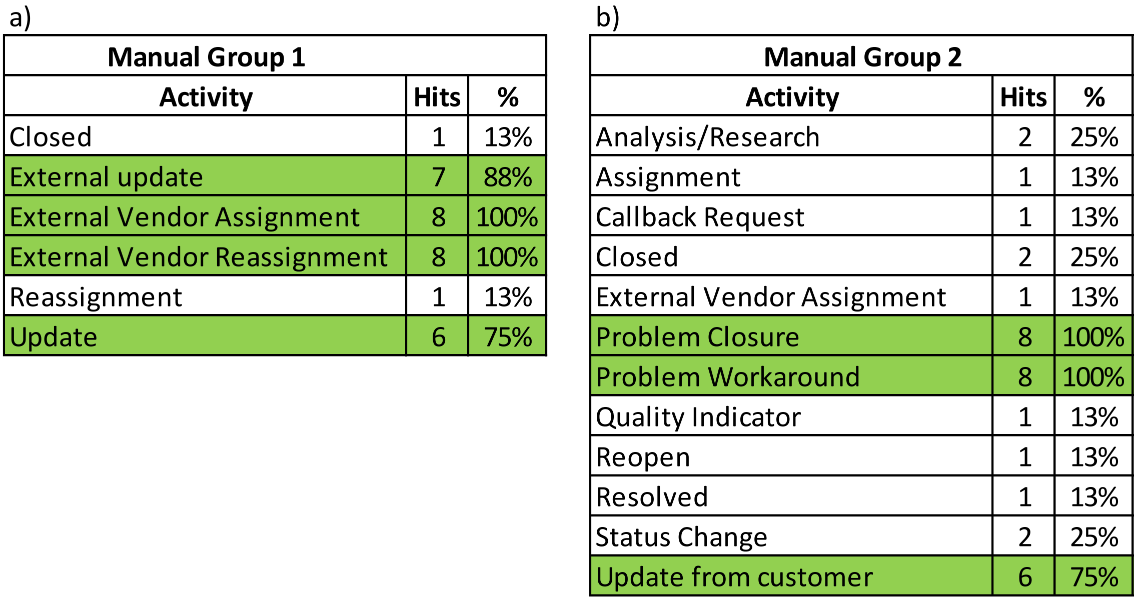
Group 5	Hits	%
Affected CI Change	1	13%
alert stage 1	1	13%
Closed	1	13%
Contact Change	8	100%
External update	1	13%
Impact Change	8	100%
Notify By Change	7	88%
Operator Update	2	25%
Service Change	8	100%
Status Change	8	100%
Update	1	13%
Urgency Change	2	25%
Vendor Reference Change	1	13%

Since 24 activities were not automatically assigned to any group, the activity list was manually inspected in order to suggest possible groups that were not considered by the automatic approach. The goal was to verify if the participants perceive any difference between an automatic or manual suggestion. Table 5.6 shows the results for the manually proposed groups. For both groups (a) and (b), it can be noticed that, again, there was a high amount of agreement on the suggested activities, above 75%, and no more than two participants agree with an inclusion of a new activity in each group. This may indicate that the automatic suggestions may be equivalent to manually created suggestions.

From the open question about the inclusion of new groups that were not previously proposed (automatic or manual), only two participants made suggestions. One participant suggested an “Incident Handling” group with the activities “Analysis/Research” and “Incident Reproduction”; and the other user made two suggestions: a “Cause analysis/research” group with “Analysis/Research” and “Incident Reproduction” activities, and an “Update follow up” group with “Description Update, Operator Update, Update from Customer” activities. These suggestions were not mentioned by any other participants.

About the possible problems on the comprehension of activity labels, Table 5.7 shows the activities were users pointed some difficulties on understanding.

Table 5.6: User’s activities selection on manually proposed groups.



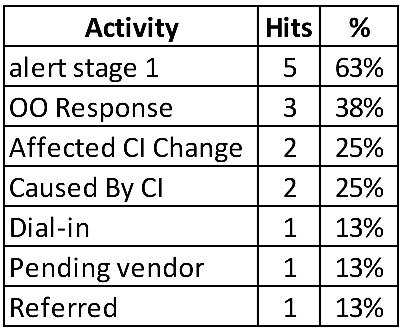
a)

Manual Group 1		
Activity	**Hits**	**%**
Closed	1	13%
External update	7	88%
External Vendor Assignment	8	100%
External Vendor Reassignment	8	100%
Reassignment	1	13%
Update	6	75%

b)

Manual Group 2		
Activity	**Hits**	**%**
Analysis/Research	2	25%
Assignment	1	13%
Callback Request	1	13%
Closed	2	25%
External Vendor Assignment	1	13%
Problem Closure	8	100%
Problem Workaround	8	100%
Quality Indicator	1	13%
Reopen	1	13%
Resolved	1	13%
Status Change	2	25%
Update from customer	6	75%

Table 5.7: Activity labels pointed as hard to understand by the participants.



Activity	Hits	%
alert stage 1	5	63%
OO Response	3	38%
Affected CI Change	2	25%
Caused By CI	2	25%
Dial-in	1	13%
Pending vendor	1	13%
Referred	1	13%

Regarding labeling quality, seven activities were marked by users as hard to understand. The activity "alert stage 1" was the most pointed (Five times checked). Two users commented: "Is there any other alert stage?" and "Why alert stage 1 if there is no 2 or 3?". The activity “OO Response” was the second with most comprehension difficulties with three indications (38%). The remaining five activities were mentioned by less than 25% of the participants. Except from "Pending vendor" that is included in Group 1, all other activities do not belong to any suggested group. This may suggest that, in this case study the poor labeling quality of these activity labels did not impact on the suggestions for groups that were automatically made. Considering the frequency of appearance in the event log only activities “Caused by CI” and “Pending Vendor” occur more than one percent, and should appear on the further resulting Declare Map. This may point to a need to refactor the activity labels on the original system in order to make clear the actions and business objects in a business process activity label.

Although the method may present problems when dealing with labeling styles and the presence of incomplete sentences that are characteristics of an activity labels in a process model (Leopold, 2013), the result of the survey showed that most of the participants agreed with the suggestions . Even when compared to manual suggestions, the results were similar. Thus, it was possible to conclude that the automatic suggestions are acceptable for this case study.

5.2.2 Case Study Results

Once the usefulness of the automatically proposed groups, considering its limitations, was confirmed by domain specialists, it was possible to generate a new Declare Map with abstractions by using the representational techniques presented in previous Chapter. Figure 5.6 shows the original Declare Map without abstractions and, for comparison, Figure 5.7 shows the resulting Declare Map with abstractions. By a visual inspection, model from Figure 5.7 has fewer elements to be analyzed, tending to be easier to understand.

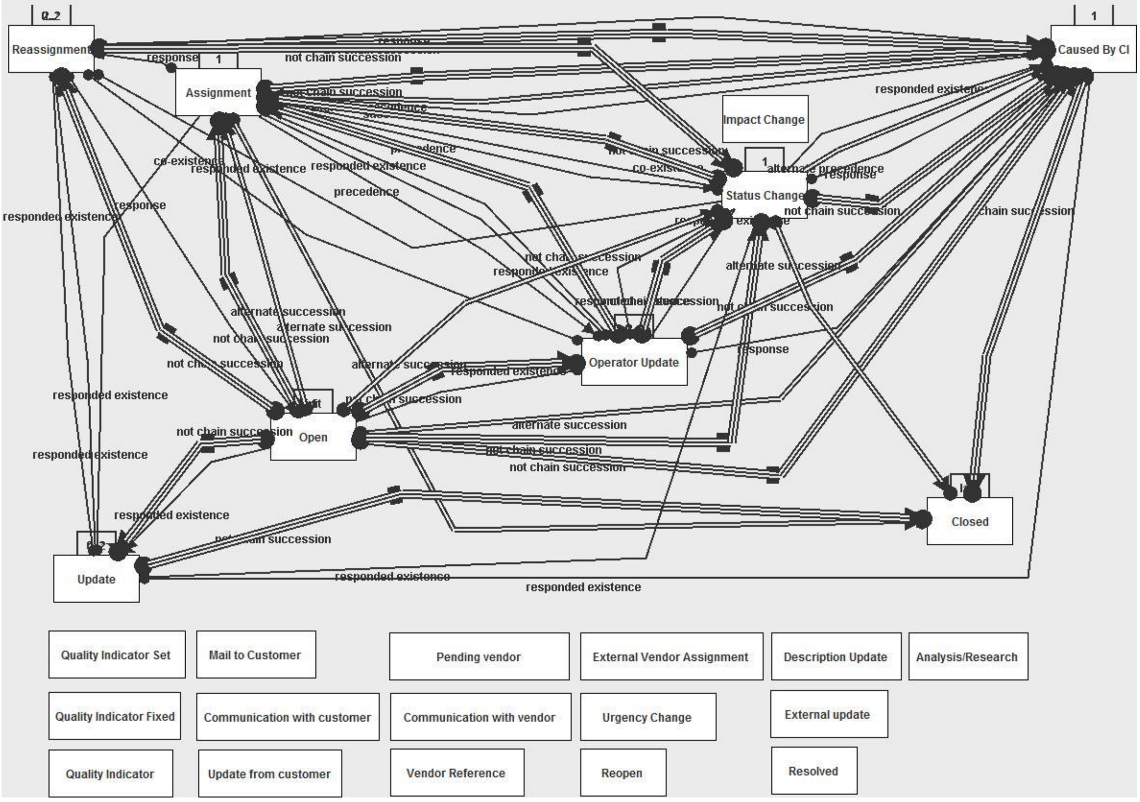


Figure 5.6: Declare Map without abstractions.

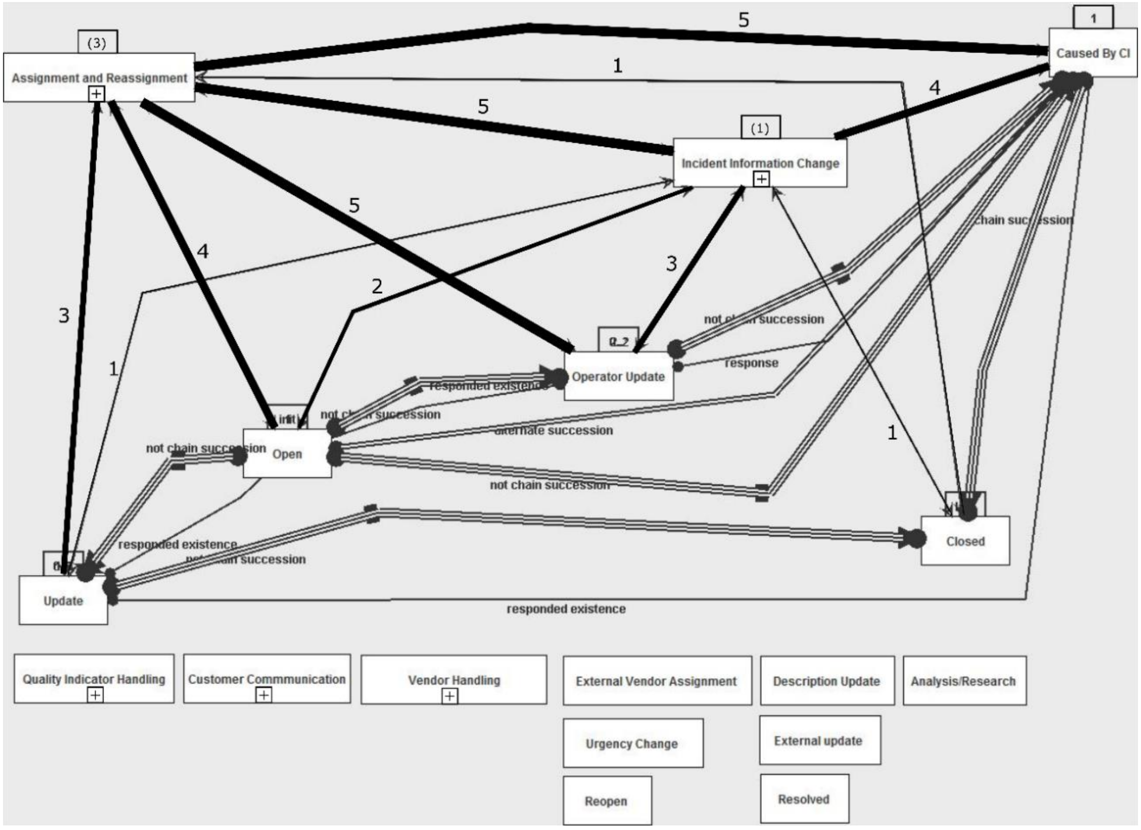


Figure 5.7: Declare map with abstractions.

From Figure 5.7, it is important to note that the aggregating arc that connects an abstract group to another activity has variable thickness that depends on the number of constraints that were abstracted. The number over the arc also indicates the number of constraints abstracted. As this representation adds a more abstract layer over the model, its original semantics is preserved. If the user wants, he or she may drill down the abstracted constraints. Figure 5.8 presents an example of this detailing. The aggregating arcs that contain multiple constraints also represent inter-groups constraints.

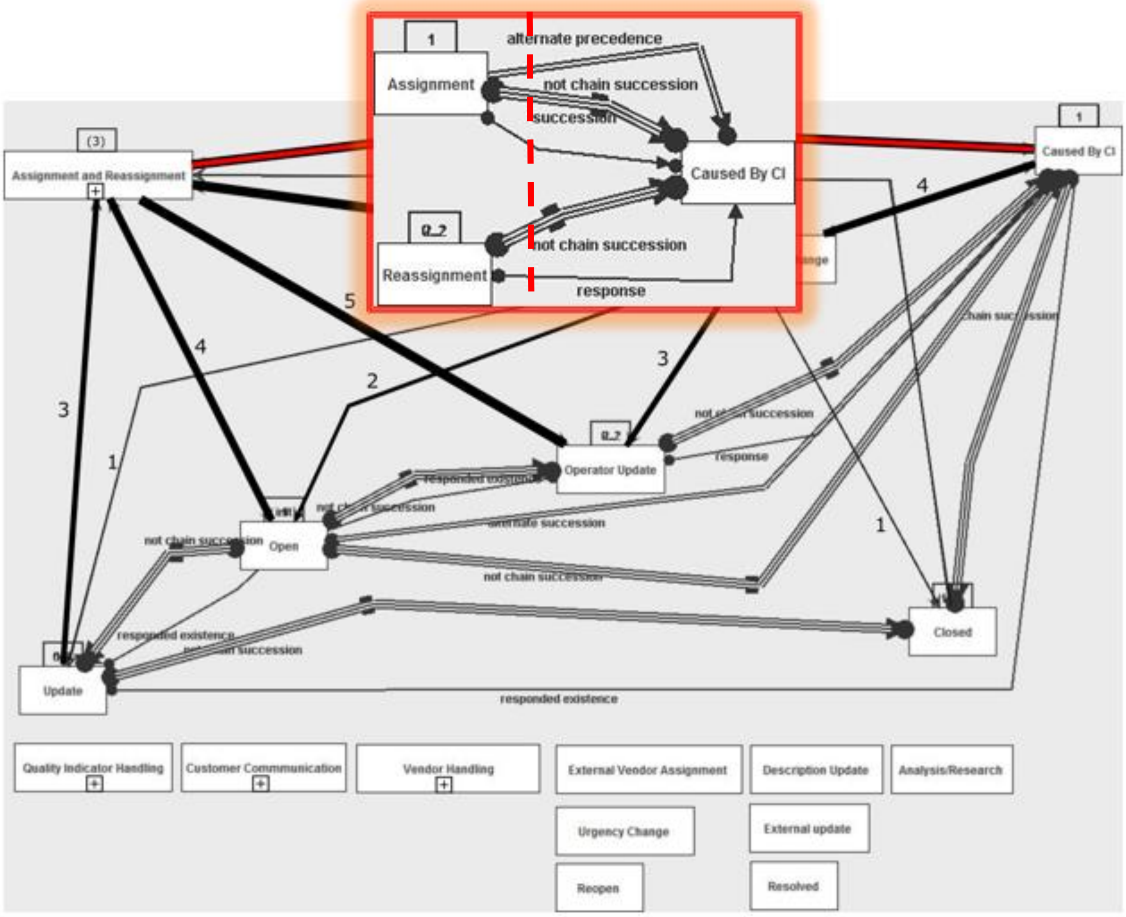


Figure 5.8: Details of the abstracted constraints between the complex activity "Assignment and Reassignment" and the activity "Caused By CI", showing inter-group constraints.

If a user wants to detail the activities belonging to a group, he or she can view them, as well as the constraints within the group, also called intra-group constraints. An example is shown in Figure 5.9.

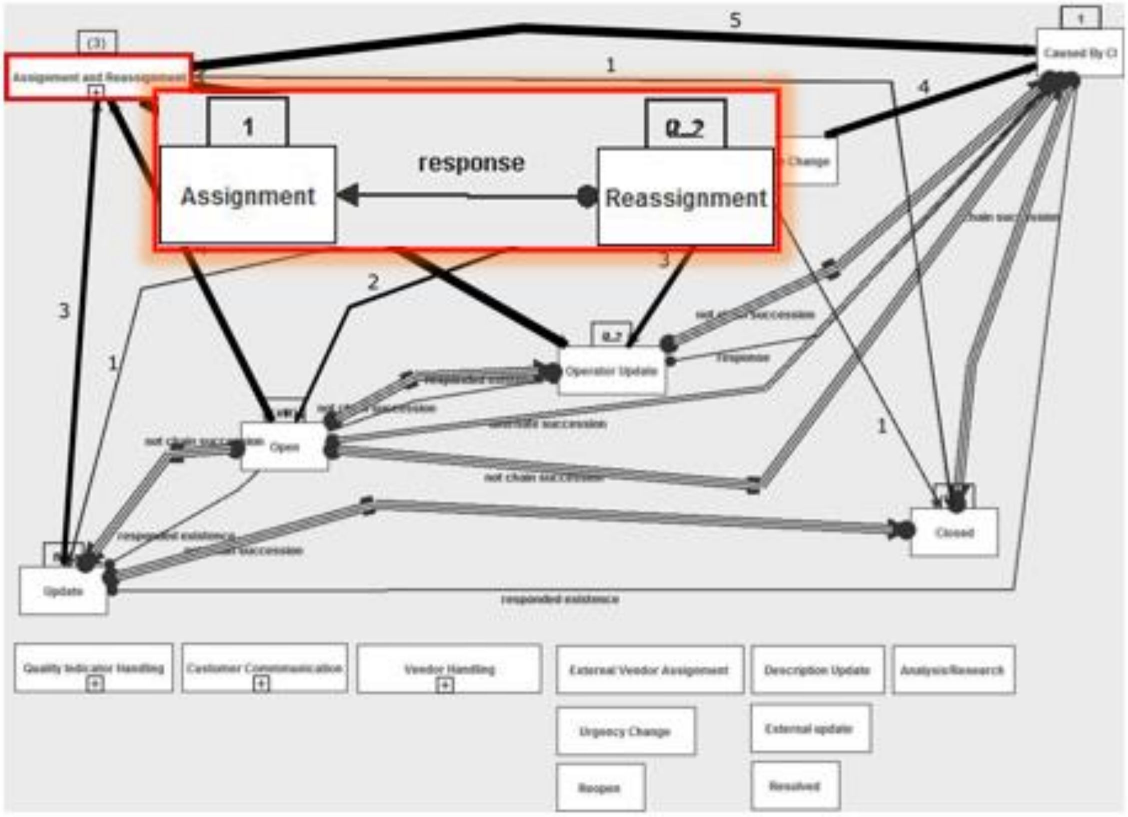


Figure 5.9: Details of the abstracted activities that belong to the "Assignment and Reassignment" complex activity.

5.3 Result Analysis

Process model's complexity metrics are useful to perform a quantitative comparison between two process models. For this analysis, complexity metrics (*Number of Constraints, Number of Activities, Number of Different Modeling Concepts, Number of Groups and Constraint-Activity Ratio)* that are applicable to declarative models, presented in Section 2.4, were selected in order to compare a Declare map without abstractions to the same model with abstractions.

The results from Table 5.8 showed that a complexity reduction was reached by using abstractions, denoted by the lower number of activities and constraints on the model with abstractions. In addition, the number of different constrains was increased by one because there was the introduction of the thickness variable arc. Also, the number of groups trivially grew due the groups proposed. The user should be aware of the risk of fragmentation (ZUGAL et al., 2013) that can require more mental effort to analyze the model when the quantity of groups is high. However, quantitative metrics about the number of activities and constraints showed significant reductions on the model with abstractions, by 32% less activities and 47% less constraints. Those metrics are related to the model size and mainly influence model's comprehensibility (MAGGI et al., 2013).

Regarding process model's quality dimensions (VAN DER AALST et al., 2012), trace fitness, precision, generalization and simplicity will remain the same as the original model, because all discovered constraints are preserved and can be verified by using available conformance checking algorithms (DE LEONI et al., 2014).

Table 5.8: Models comparison by complexity metrics.

	Model without Abstractions	*Model with Abstractions*	*Reduction Percentage*
Nº of Activities	25	17	32%
Nº of Constraints	59	31	47%
Different modeling concepts	15	16	-7%
Nº of Groups	0	5	-
Constraint/Activity Ratio	2.36	1.82	23%

5.4 Final Remarks

As observed in this case study, the proposed method generated a suitable set of groups of activities that could be used to create abstractions that made a Declare Map less complex and tending to be easier to understand. However, this conclusion is restricted to this case study and may not be generalized to different process models without further evaluations. As there are few real life event logs publicly available, and even fewer that gather characteristics for declarative modelling, the current analysis is limited to this case study. The groups' evaluation by specialists was made by professionals with different levels of knowledge on the IT Service Management domain, and was limited to the number of participants. These facts may introduce some subjectivity on user's evaluation and results may be different on other scenarios, with different process models and different people evaluating the groups. The choice of asking domain specialists instead of people in general is because the event log came from an ITSM system that is domain specific. Therefore, repeating this case study on different scenarios and evaluating them with a higher number of domain specialists are important steps considered for future work. Another futher improvement is to expand the experiment by introducing control groups, manually designed to contain unrelated activities. This kind of groups would help identifying any bias caused by the participants, due to fatigue, the order of the survey, or demand characteristics (ORNE, 1962).

As a contribution for declarative models analysis, the proposed method suggested groups of related activities that may be used to create a visualization layer that abstracts some details and helps user on understanding a declarative business process model in a higher abstraction level.

Chapter 6 - Conclusion

This chapter presents the main features from this thesis and points its contributions, limitations and possibilities of future work for research continuity.

In a declarative models discovery scenario, there is a need to deal with complexity caused by the high number of discovered constraints, which hinders user understandability and impose difficulties on model analysis. For traditional imperative process models, the use of abstractions successfully addressed this problem (SMIRNOV et al, 2011; LI et al, 2011; GÜNTHER et al, 2007). However, there are few works that deal with abstractions on declarative models (ZUGAL et al, 2013; DEBOIS et al, 2014). This dissertation aimed to contribute to declarative models discovery by presenting a novel technique to suggest abstract views for a low-level declarative model, when no information beyond activity labels is available. Those abstract views are derived from a linguistic analysis and clustering of process activity labels.

The proposal was assessed through a case study with a real life dataset from the Business Process Intelligence Challenge 2014, which has the characteristics of an unstructured business process. After applying the proposed method and having domain experts evaluate the suggested groups, the creation of an abstraction layer over the discovered declarative model implied on a reduction of 32% on the number of activities, and 47% on the number of constraints presented to the user, pointing to a complexity reduction when compared to the model without abstractions.

6.1 Contributions

The main contribution of this work is the application of a novel semantic abstraction criterion on declarative models in order to cope with complexity problems resulting for process mining. The technical solution includes the formulation of two algorithms to suggest groups of activity labels on declarative process models, in a user guided fashion. A prototypical software was built to operationalize the algorithms and automatically suggest activity groups. Besides the semantic abstraction criterion, we proposed a method for creating abstract process views composed by the suggested groups of activity labels over a low-level declarative map. The application of the method in a real life environment performed by a case study confirmed that the method is able to reduce the complexity of an automatically discovered declarative model and produces useful groups for domain stakeholder's analysis.

In addition to this thesis, for expanding the body of knowledge, this research also produced two scientific papers presented in an international and a national conference respectively (RICHETTI et al., 2014a and 2014b).

6.1.1 Implications for research

As main implications for research, the proposed method can be applied on declarative process modeling and process mining disciplines. With regard to the applicability, the proposed approach is also applicable to any declarative modeling language, such as Declare, DCR Graphs or ReFlex. In general, the proposed visual notation can be integrated with any constraint-based declarative modeling language with a graphical notation. In consequence, these models can be enriched with more abstract information and a drill down mechanism for the analysis of details.

For process mining, reducing the complexity of a flexible or unstructured process model is still a concern. Several techniques have been proposed to produce less complex declarative models (BOSE et al., 2013; MAGGI et al., 2013; MAGGI et al., 2012). In another research branch, there exist works explaining how to model hierarchy and subprocesses in declarative processes models (ZUGAL et al., 2013; DEBOIS et al., 2014), but few approaches try to apply these concepts on process mining scenarios. In such scenarios, sometimes the results present very complex models and the automatic support for creating abstract views helps on model's analysis. The approach of this thesis complements previous techniques by addressing the semantic dimension of process activity labels to create abstractions that are key for complexity reduction.

As secondary goals, the semantic similarity measure between activities may also be applied on process model matching, since this area also considers pairwise comparisons to evaluate how process models differ among each other. Finally, the proposed grouping strategy can also be used for the creation of clusters of semantically similar elements on different areas beyond BPM, such as the creation of sense clusters for ranking abbreviations expansions present in document collections (BRACEWELL et al., 2005).

6.1.2 Implication for practice

The results of this thesis also have implications for practice. The techniques may be used in process mining scenarios in collaboration with existing tools. They can help users on the creation of less complex and more abstract declarative models after discovering a new process model, even when there is no previous knowledge available about the process model, which is a situation that could happen on such scenarios. Moreover, the proposed techniques can be integrated into commercial modeling tools in order to support modelers by automatically suggesting groups after the design of a model or to improve an existing process model.

6.2 Limitations

From the implementation, there are performance issues that need to be considered before creating a new software system or embedding the techniques into an existing tool. The extensive use of semantic similarity calculations at word-level affects algorithm's performance, due the necessity of traversing lexical databases multiple times. For the presented case study, with the specified hardware and an event log with more than four hundred thousand events, the algorithm took about four hours to run completely, an execution time that prevents the method to be used in an online analysis.

The labeling quality is crucial for generating useful information when creating groups of activities. Poor labeling may negatively impact groups suggestions. To avoid this limitation and improve results, it is important to follow recommended guidelines for labeling process activities (LEOPOLD, 2013). Regarding the creation of groups of activities, manual grouping usually considers more information beyond the semantics, which can be biased due to personal experiences and judgment of the modeler. It is not trivial to make a system capable of considering the same contextual information when performing the creation of a group as a person does. However, the use of semantic information from the activity labels showed a good acceptance among domain experts in the case study. Thus, the proposed approach intends to help domain specialists or beginner practitioners to have a better understanding on declarative process models.

Generalization of this solution is limited by the single case study presented. For a more robust assessment on the quality of suggestions for groups of activities, there is a need to evaluate the method with different datasets. The number of participants that performed the analysis limited the evaluation of the groups' suggestions. Thus, for broadening the application possibility to other scenarios, it is important to conduct new groups' evaluations with novel datasets and with a higher number of domain experts.

6.3 Future Work

For future work, there are plans to consider abstractions in full conformance with subprocess rules for declarative models stated in (ZUGAL, 2013), also considering temporal aspects, which can turn the abstraction layers executable. Another possibility is to embed this method on process modelling tools in order to provide support for creating abstract views from low-level designed processes or to help maintain process model collections. For this, the development of a ProM plugin to generate the visualization layers for discovered declarative models using this framework is suitable. Another possibility is to embed the algorithms on the DCRgraphs.net[19] modeling and execution tool to help modelers on automatically retrieving suggestions for subprocesses.

In addition to the development issues, there are some opportunities to optimize the algorithm in order to reduce its execution time, which should be tested in order to assess performance limits. The generation of groups suggestions may also be further investigated by directly analyzing the discovered process model, instead of the event log. If the execution time is reduced, the method may be executed on line during process mining, when the proposed algorithm may be executed more than once during the iterative analysis and tuning of the process mining output.

The semantic similarity between activity labels may be extended by using concepts from sentence similarity calculation. Even knowing that activity labels are more likely to be incomplete sentences and they have identifiable structural elements, such as actions and objects (LEOPOLD, 2013), it will be interesting to consider aspects such as combining word overlap, term frequency and linguistic measures applied on computing sentence similarity (ACHANANUPARP et al., 2008), beyond the semantic abstraction and aggregation aspects.

Regarding the subprocess metrics applied on the analysis, a single metric may be developed combining precision, recall, undershoot and overshoot.

[19] See http://dcrgraphs.net.

To cope with the generalization limitations, new case studies must be performed, expanding the survey with improved control groups, also using different datasets and more domain experts involved.

References

ABECKER, A., BERNARDI, A., ESLT. L. V., et al., "Workflow-embedded organizational memory access: The DECOR project". *IJCAI'2001 Working Notes of the Workshop on Knowledge Management and Organizational Memories, Seattle, Washington, USA*, pp. 1-9, 2001.

ACHANANUPARP, P., HU, X., SHEN, X. "The evaluation of sentence similarity measures". In: *Data Warehousing and Knowledge Discovery*. Springer Berlin Heidelberg, pp. 305-316, 2008.

BAIER, T., MENDLING, J. "Bridging Abstraction Layers In Process Mining By Automated Matching of Events and Activities". In: *Business Process Management*, pp. 17-32. Springer, 2013.

BAYER, K., KEMPF, S., BROCKS, H., et al. "A Multiagent Environment for The Flexible Enactment of Knowledge-Intensive Processes". *Cybernetics and Systems: An International Journal*, *37*(6), pp. 653-672, 2006.

BECKER, J., ROSEMANN, M., VON UTHMANN, C. "Guidelines of business process modeling". In: *Business Process Management*. Springer, pp. 30-49, 2000.

BOSE, R. P. J. C., VERBEEK, E. H. M. W., VAN DER AALST, W. M. P. "Discovering Hierarchical Process Models Using ProM". In: *Nurcan, S. (eds.) IS Olympics: Information Systems in a Diverse World*, LNBIP, vol. 107, pp. 33-48. Springer Berlin Heidelberg, 2012.

BOSE, R.P.J.C., MAGGI, F.M., VAN DER AALST, W.M.P. "Enhancing Declare Maps Based on Event Correlations". In: *Business Process Management*, pp. 97-112 Springer, 2013.

BRACEWELL, D.B., RUSSELL, S., WU, A. S. "Identification, expansion, and disambiguation of acronyms in biomedical texts". In: *Parallel and Distributed Processing and Applications Workshops*. Springer, pp. 186-195, 2005.

BRIN, S., MOTWANI, R., SILVERSTEIN, C. "Beyond Market Baskets: Generalizing Association Rules to Correlations". *ACM SIGMOD Record*. Vol. 26. No. 2. ACM, 1997.

BRON, C., KERBOSCH, J. “Algorithm 457: finding all cliques of an undirected graph”. *Communications of the ACM*, v. 16, n. 9, pp. 575-577, 1973.

CARDOSO, J., MENDLING, J., NEUMANN, G., et al. “A discourse on complexity of process models”. In: *Business process management workshops,* pp. 117-128. Springer Berlin Heidelberg, 2006.

CARVALHO, R. M. D., SILVA, N. C., LIMA, R. M., et al. “Reflex: an efficient graph-based rule engine to execute declarative processes”. In: *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*, pp. 1379-1384. IEEE, 2013.

CAYOGLU, U., DIJKMAN, R.M., DUMAS, M., et al. “The Process Model Matching Contest 2013”. In: *4th International Workshop on Process Model Collections: Management and Reuse (PMC-MR'13)*, 2013.

DAVENPORT T.H. “Thinking for a Living: How to Get Better Performance and Results from Knowledge Workers”. Harvard Business Rev. Press, 2005.

DAVENPORT, T.H., Short, J.E. “The new industrial engineering: Information technology and business process redesign”. *Sloan Management Review* 31(4), pp. 11–27, 1990.

DE LEONI, M., MAGGI, F.M., VAN DER AALST, W.M.P. “An alignment-based framework to check the conformance of declarative process models and to preprocess event-log data”. *Information Systems*, 2014.

DEBOIS, S., HILDEBRANDT, T., SLAATS, T. “Hierarchical declarative modelling with refinement and sub-processes”. In: *Business Process Management. Springer International Publishing*, pp. 18-33, 2014.

DI CICCIO, C., MECELLA, M. “A Two-Step Fast Algorithm for the Automated Discovery of Declarative Workflows”. *Computational Intelligence and Data Mining*, IEEE, 2013.

DI CICCIO, C., MARRELLA, A., RUSSO, A. “Knowledge-intensive Processes: An Overview of Contemporary Approaches”. *Knowledge-intensive Business Processes*, pp. 33, 2012.

DI CICCIO, C., MECELLA, M. “Studies on the discovery of declarative control flows from error-prone data”. *Data-Driven Process Discovery and Analysis SIMPDA 2013*, pp. 31, 2013.

DUMAS, M., VAN DER AALST, W.M.P., TER HOFSTEDE, A.H. *Process-Aware Information Systems: Bridging People and Software Through Process Technology*. John Wiley & Sons, 2005.

DUMAS, M, LA ROSA, M, MENDLING, J., et al. *Fundamentals of business process management*. Heidelberg: Springer, 2013.

ESHUIS, R., GREFEN, P. “Constructing Customized Process Views”. *Data & Knowledge Engineering*, v. 64, n. 2, pp. 419-438, 2008.

FAHLAND, D., LÜBKE, D, MENDLING, J., et al. “Declarative Versus Imperative Process Modeling Languages: the Issue of Understandability”. *Enterprise, Business-Process and Information Systems Modeling*, pp. 353-366. Springer, 2009.

FAN, T., CHANG, C. “Sentiment-oriented contextual advertising”. *Knowledge and Information Systems*, v. 23, n. 3, pp. 321-344, 2010.

FRAKES, W. B., BAEZA-YATES, R. *Information Retrieval, Data Structure and Algorithms*. Prentice Hall, 1992.

FRANÇA, J.B.S. *Uma Ontologia para Definição de Processos Intensivos em Conhecimento*, Dissertação de Mestrado. Departamento de Informática Aplicada, UNIRIO, Rio de Janeiro, RJ, Brasil, 2012.

GÜNTHER, C.W., VAN DER AALST, W.M.P. “Fuzzy Mining - Adaptive Process Simplification Based on Multi-perspective Metrics”. In: *Business Process Management*. Volume 4714 of LNCS, pp. 328–343, 2007.

HAISJACKL, C., ZUGAL, S., SOFFER, P., et al. “Making Sense of Declarative Process Models: Common Strategies and Typical Pitfalls”. *Enterprise, Business-Process and Information Systems Modeling*, pp. 2-17. Springer, 2013.

HAREL, D., PNUELI, A., SCHMIDT, J. P., SHERMAN, S. “On the formal semantics of statecharts”. In: *Proceedings of the Second IEEE Symposium on Logic in Computation*, pp. 54-64. IEEE, 1987.

HILDEBRANDT, T., MUKKAMALA, R.R. “Declarative event-based workflow as distributed dynamic condition response graphs”. In: *PLACES*. EPTCS, vol. 69, pp. 59–73, 2010.

HILDEBRANDT, T., MUKKAMALA, R.R., SLAATS, T. “Nested dynamic condition response graphs”. In: “Fundamentals of Software Engineering”. Springer Berlin Heidelberg, pp. 343-350, 2012.

HUCKVALE, T., OULD, M. “Process Modelling - Who, What and How: Role Activity Diagramming”. In: *Business process change: Reengineering concepts, methods and technologies*. IGI Global, pp. 330-349, 1995.

KLINKMÜLLER, C., WEBER, I., MENDLING, J., et al. "Increasing recall of process model matching by improved activity label matching". In: *Business Process Management*. Springer Berlin Heidelberg, pp. 211-218, 2013.

KODRATOFF, Y. "Knowledge discovery in texts: A definition and applications". *Lecture Notes in Computer Science,* 1609, pp. 16–29, 1999.

LA ROSA, M., WOHED, P., MENDLING, J., et al. "Managing Process Model Complexity via Abstract Syntax Modifications. Industrial Informatics". *IEEE Transactions* on, v. 7, n. 4, pp. 614-629, 2011.

LAUE, R., MENDLING, J. "Structuredness and Its Significance for Correctness of Process Models". *Information Systems and E-Business Management,* v. 8, n. 3, pp. 287-307, 2010.

LEOPOLD, H., MENDLING, J., HEIJERS, H.A., et al. "Simplifying process model abstraction: Techniques for Generating Model Names". *Information Systems 39,* pp. 134-151, 2014.

LEOPOLD, H. *Natural language in business process models*. Springer, 2013.

LI, J., BOSE, R.P.J.C., VAN DER AALST, W.M.P. "Mining Context-Dependent and Interactive Business Process Maps Using Execution Patterns". In: *Business Process Management Workshops*. Springer Berlin Heidelberg, 2011.

LIN, D. "An Information-Theoretic Definition of Similarity". In: *Proceedings of the 15th International Conference on Machine Learning*, pp. 296–304, 1998.

LINDLAND, O. I., SINDRE, G., SOLVBERG, A. "Understanding quality in conceptual modeling". *Software, IEEE*, v. 11, n. 2, pp. 42-49, 1994.

MAGGI, F.M, MOOIJ, A.J., VAN DER AALST, W.M.P. "User-guided Discovery of Declarative Process Models". *Computational Intelligence and Data Mining,* IEEE, 2011.

MAGGI, F.M., BOSE, R.P.J.C., VAN DER AALST, W.M.P. "A Knowledge-Based Integrated Approach for Discovering and Repairing Declare Maps". *Advanced Information Systems Engineering*. Springer Berlin Heidelberg, 2013.

MAGGI, F.M., BOSE, R.P.J.C., VAN DER AALST, W.M.P. "Efficient Discovery of Understandable Declarative Process Models from Event Logs". *Advanced Information Systems Engineering*. Springer Berlin Heidelberg, 2012.

MAGGI, F.M., SLAATS, T., REIJERS, H.A. "The automated discovery of hybrid processes". In: *Business Process Management*. Springer, pp. 392-399, 2014.

MALDONADO, M. U.: *Análise do impacto das políticas de criação e transferência de conhecimento em processos intensivos em conhecimento: Um modelo de dinâmica de sistemas*. Dissertação de Mestrado. Programa de Pós-Graduação em Engenharia e Gestão do Conhecimento, Universidade Federal de Santa Catarina, Florianópolis, SC, Brasil, 2008.

MAYER, R. J., MENZEL, C. P., PAINTER, M. K., et al. "Information Integration for Concurrent Engineering (IICE) - IDEF3 Process Description Capture Method Report". Technical Report, 1995.

MENDLING, J., REIJERS, H.A., CARDOSO, J. "What makes process models understandable?". In: *Business Process Management*. Springer, pp. 48-63, 2007.

MICHELIZZI, J., PEDERSEN, T. "WordNet–SenseRelate–WordToSet-0.02". Available at search.cpan.org/dist/WordNet-SenseRelate-WordToSet, 2005.

MOODY, D. L. "Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions". *Data & Knowledge Engineering*, v. 55, n. 3, pp. 243-276, 2005.

NACHIAPPAN, N., BALL, T., ZELLER, A. "Mining metrics to predict component failures". In: *Proceedings of the 28th International Conference on Software Engineering*, Shanghai, China, 2006.

OMG. Object Management Group, Unified Modelling Language, 2005. Disponível em: http://www.uml.org/. Último acesso abril 2015.

OMG. Object Management Group, Business Process Modelling Notation, 2011. Disponível em: http://www.omg.org/spec/BPMN/2.0/. Último acesso abril 2015.

ORNE, Martin T. "On the social psychology of the psychological experiment: With particular reference to demand characteristics and their implications". *American psychologist*, v. 17, n. 11, p. 776, 1962.

PEDERSEN, T. "Information content measures of semantic similarity perform better without sense-tagged text." *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010.

PETERSON, J.L. *Petri net theory and the modeling of systems*. New Jersey: Prentice Hall, 1981.

PICHLER, P., WEBER, B., ZUGAL, S., et al. "Imperative Versus Declarative Process Modeling Languages: an Empirical Investigation". In: *Business Process Management Workshops*. Springer Berlin Heidelberg, pp. 383-394, 2012.

POLYVYANYY, A., SMIRNOV, S., WESKE, M. “Business Process Model Abstraction”. In: *Handbook on Business Process Management 1*. Springer Berlin Heidelberg, pp. 147-165, 2015.

PORTER, M.F. *Readings in information retrieval*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 313-316, 1997.

RECKER, J. *Scientific Research in Information Systems: A Beginner’s Guide (Progress in IS)*. Springer, 2012.

REIJERS, H.A., MENDLING, J., Dijkman, R.M. *On the usefulness of subprocesses in business process models*. External Report, BPM center report, No. BPM-10-03. Eindhoven: BPMcenter.org, 2010.

REIJERS, H.A., SLAATS, T., STAHL, C. “Declarative Modeling–An Academic Dream or the Future for BPM?”. In: *Business Process Management*. Springer Berlin Heidelberg, pp. 307-322, 2013.

RESNIK, P. “Using information content to evaluate semantic similarity in a taxonomy”. In: *Proceedings of IJCAI-95*, pp. 448–453, Montreal, Canada, 1995.

RICHETTI, P.H.P., BAIÃO, F.A., SANTORO, F.M. “Declarative Process Mining: Reducing Discovered Models Complexity by Pre-Processing Event Logs”. In: *Business Process Management*, pp. 400-407. Springer, 2014.

RICHETTI, P.H.P., BAIÃO, F.A., SANTORO, F.M. “Descoberta de Modelos de Processos Declarativos: Uma Abordagem para Redução de Complexidade com Ênfase no Pré-processamento”. *In: WTDSI - X SBSI*. pp. 55-57. 2014.

RISTAD, E.S., YIANILOS, P.N. “Learning string-edit distance”. *Pattern Analysis and Machine Intelligence, IEEE Transactions* on, v. 20, n. 5, pp. 522-532, 1998.

SCHEER, A.W. *ARIS - Business process frameworks*. Springer, 1999.

SMIRNOV, S., DIJKMAN, R., MENDLING, J., et al. “Meronymy-Based Aggregation of Activities in Business Process Models”. *Conceptual Modeling–ER 2010*, pp. 1-14. Springer Berlin Heidelberg, 2010.

SMIRNOV, S., REIJERS, H.A., WESKE, M. “A Semantic Approach For Business Process Model Abstraction”. In: *Advanced Information Systems Engineering*. Springer Berlin Heidelberg, pp. 497-511, 2011.

SMITH, J.M., SMITH, D.C.P. "Database Abstractions: Aggregation and Generalization." *ACM Transactions on Database Systems (TODS) 2.2*, pp. 105-133, 1977.

SOFFER, P., ROLLAND, C. “Combining intention-oriented and state-based process modeling”. In: *Conceptual Modeling–ER 2005*. Springer Berlin Heidelberg, pp. 47-62, 2005.

LEVENSHTEIN, V. I. “Binary codes capable of correcting deletions, insertions and reversals”. *Soviet Physics Doklady*., 10(8), pp. 707–710, 1966.

VAN DER AALST, W., ADRIANSYAH, A., DE MEDEIROS, A. K. A., et al. “Process mining manifesto”. In: *Business Process Management Workshops*. Springer Berlin Heidelberg, pp. 169-194, 2012.

VAN DER AALST, W.M.P. "Business Process Management: a Comprehensive Survey." *ISRN Software Engineering 2013*, 2013.

VAN DER AALST, W.M.P. *Discovery, Conformance and Enhancement of Business Processes*. Springer, Heidelberg, 2011.

VAN DER AALST, W.M.P., PESIC, M., SCHONENBERG, H. “Declarative workflows: Balancing between flexibility and support”. *Computer Science-Research and Development*, v. 23, n. 2, pp. 99-113, 2009.

VAN DER AALST, W.M.P., TER HOFSTEDE, A.H.M., WESKE, M. “Business process management: A survey”. In: *Business Process Management*. Springer Berlin Heidelberg, pp. 1-12, 2003.

VAN DER AALST, W.M.P. *Process mining: discovery, conformance and enhancement of business processes*. Springer Science & Business Media, 2011.

VAN DONGEN, B.F. “BPI Challenge 2014: Incident details”. Rabobank Dataset. http://dx.doi.org/10.4121/uuid:3cfa2260-f5c5-44be-afe1-b70d35288d6d, 2014.

WEBER, B., REICHERT, M., MENDLING, J., et al.: “Refactoring Large Process Model Repositories”. *Computers in Industry* 62.5, pp. 467-486, 2011.

WEIDLICH, M., SHEETRIT, E., BRANCO, M. C., et al. “Matching business process models using positional passage-based language models”. In: *Conceptual Modeling*. Springer Berlin Heidelberg, pp. 130-137, 2013.

WESTERGAARD, M., STAHL, C., REIJERS, H.A. *UnconstrainedMiner: efficient discovery of generalized declarative process models*. BPM Center Report BPM-13-28, BPMcenter. org, pp. 28, 2013.

WU, X., ZHANG, C., ZHANG, S. “Efficient Mining of Both Positive and Negative Association Rules”. *ACM Transactions on Information Systems (TOIS) 22.3*, pp. 381-405, 2004.

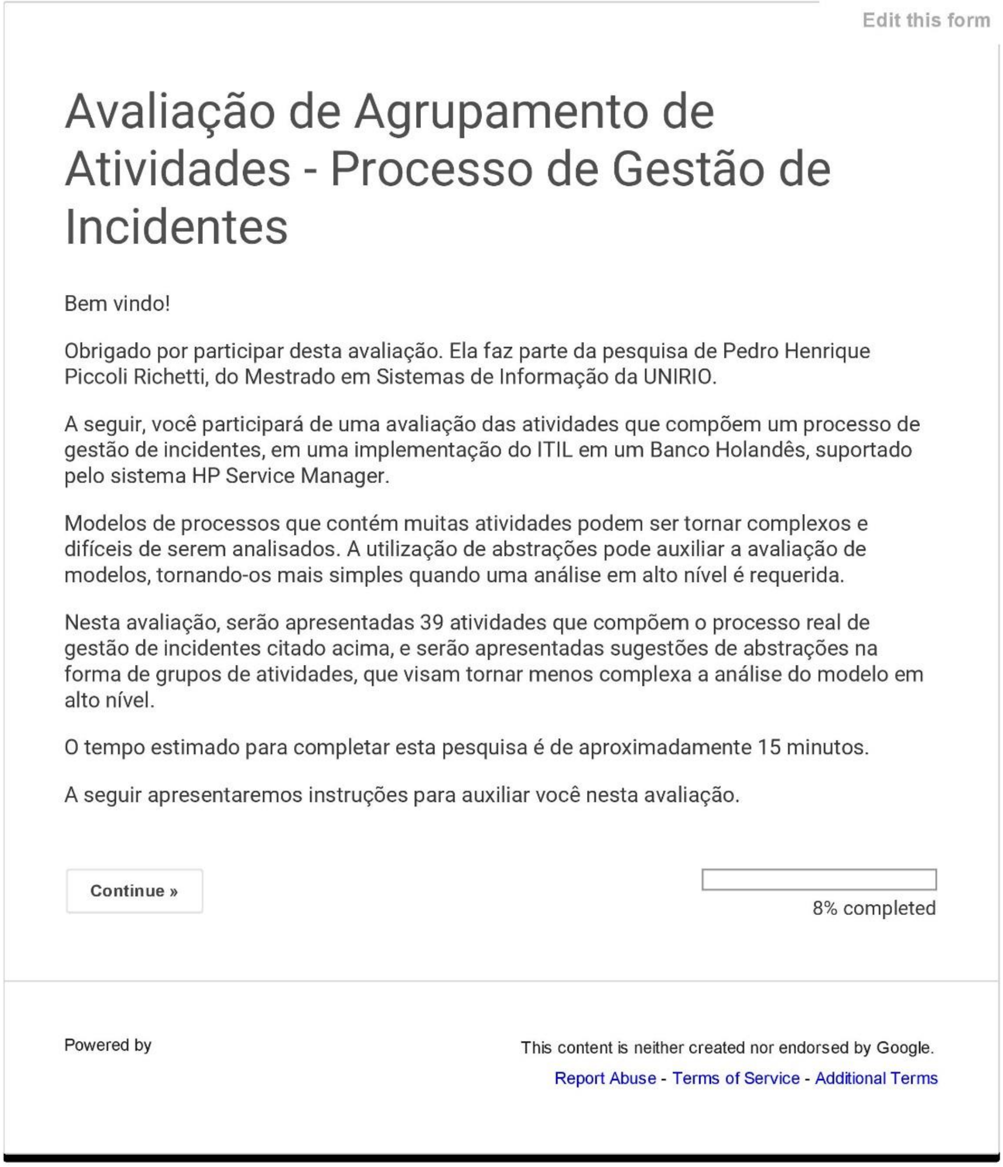
WU, Z., PALMER, M. “Verb semantics and lexical selection”. In: *Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics*, pp. 133–138, Las Cruces, New Mexico, 1994.

ZUGAL, S., SOFFER, P., HAISJACKL, C., et al. “Investigating Expressiveness and Understandability of Hierarchy in Declarative Business Process Models”. *Software & Systems Modeling*, pp. 1-23, 2013.

ZUGAL, S., PINGGERA, J., MENDLING, et al. “Assessing the impact of hierarchy on model understandability – A cognitive perspective”. In: *Proceedings of the EESS-Mod ’11*, pp. 123–133, 2011.

ZUR MUEHLEN, M., SWENSON, K.D., “BPAF: A Standard for the Interchange of Process Analytics Data (September 3, 2010)”. *Business Process Management Workshops: BPM 2010 International Workshops and Education Track*, Hoboken, NJ, USA, September 13-15, 2010.

Appendix I – Online Survey to Evaluate Groups’ Suggestions



Avaliação de Agrupamento de Atividades - Processo de Gestão de Incidentes

Bem vindo!

Obrigado por participar desta avaliação. Ela faz parte da pesquisa de Pedro Henrique Piccoli Richetti, do Mestrado em Sistemas de Informação da UNIRIO.

A seguir, você participará de uma avaliação das atividades que compõem um processo de gestão de incidentes, em uma implementação do ITIL em um Banco Holandês, suportado pelo sistema HP Service Manager.

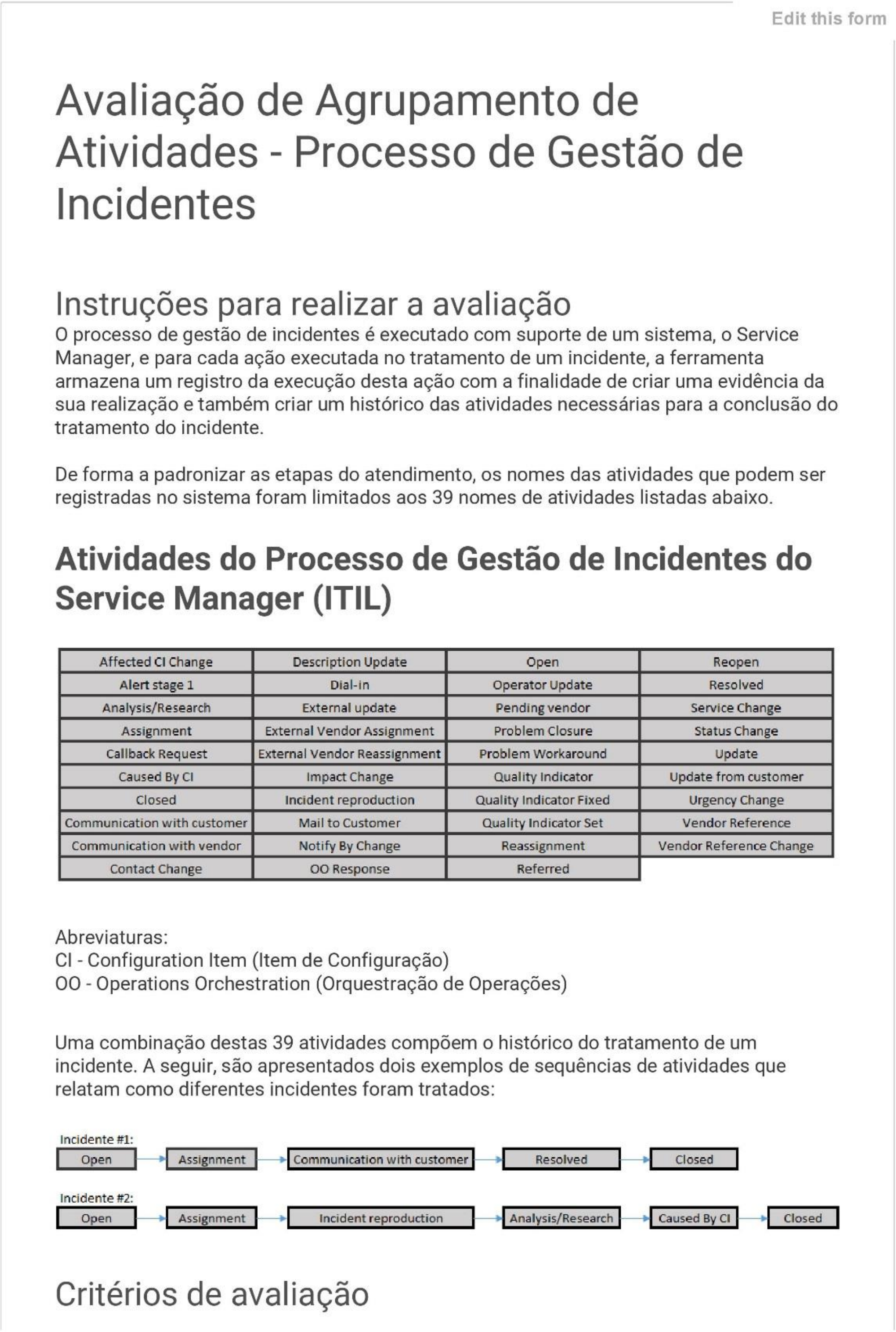
Modelos de processos que contém muitas atividades podem ser tornar complexos e difíceis de serem analisados. A utilização de abstrações pode auxiliar a avaliação de modelos, tornando-os mais simples quando uma análise em alto nível é requerida.

Nesta avaliação, serão apresentadas 39 atividades que compõem o processo real de gestão de incidentes citado acima, e serão apresentadas sugestões de abstrações na forma de grupos de atividades, que visam tornar menos complexa a análise do modelo em alto nível.

O tempo estimado para completar esta pesquisa é de aproximadamente 15 minutos.

A seguir apresentaremos instruções para auxiliar você nesta avaliação.

This content is neither created nor endorsed by Google.



Edit this form

Avaliação de Agrupamento de Atividades - Processo de Gestão de Incidentes

Instruções para realizar a avaliação

O processo de gestão de incidentes é executado com suporte de um sistema, o Service Manager, e para cada ação executada no tratamento de um incidente, a ferramenta armazena um registro da execução desta ação com a finalidade de criar uma evidência da sua realização e também criar um histórico das atividades necessárias para a conclusão do tratamento do incidente.

De forma a padronizar as etapas do atendimento, os nomes das atividades que podem ser registradas no sistema foram limitados aos 39 nomes de atividades listadas abaixo.

Atividades do Processo de Gestão de Incidentes do Service Manager (ITIL)

Affected CI Change	Description Update	Open	Reopen
Alert stage 1	Dial-in	Operator Update	Resolved
Analysis/Research	External update	Pending vendor	Service Change
Assignment	External Vendor Assignment	Problem Closure	Status Change
Callback Request	External Vendor Reassignment	Problem Workaround	Update
Caused By CI	Impact Change	Quality Indicator	Update from customer
Closed	Incident reproduction	Quality Indicator Fixed	Urgency Change
Communication with customer	Mail to Customer	Quality Indicator Set	Vendor Reference
Communication with vendor	Notify By Change	Reassignment	Vendor Reference Change
Contact Change	OO Response	Referred	

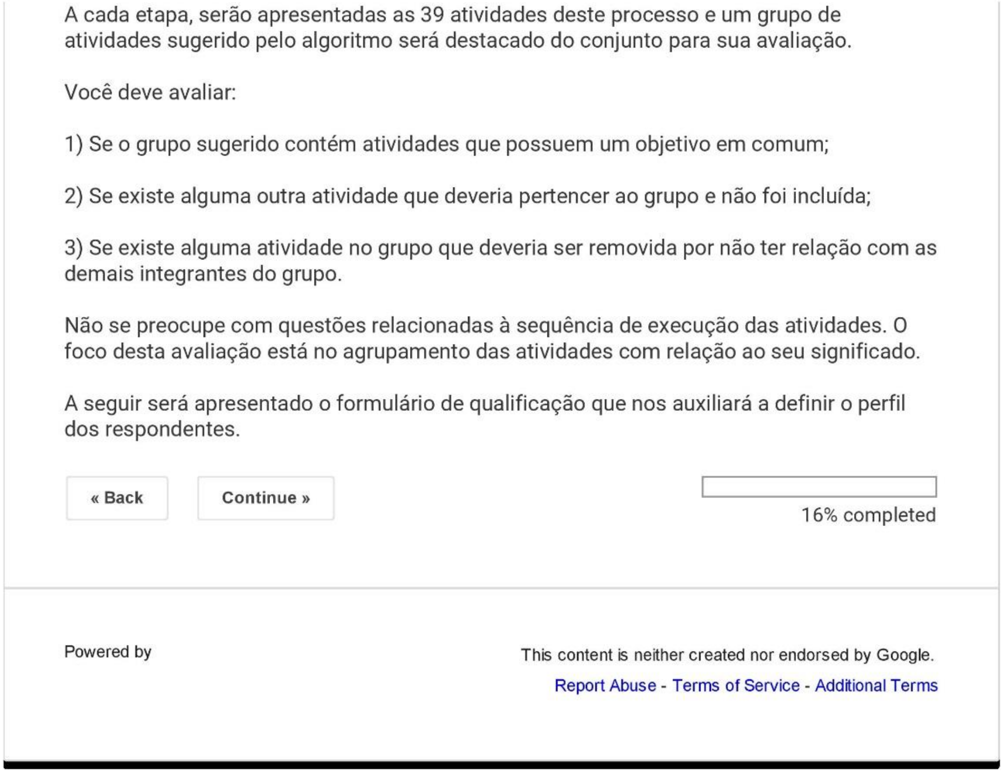
Abreviaturas:
CI - Configuration Item (Item de Configuração)
OO - Operations Orchestration (Orquestração de Operações)

Uma combinação destas 39 atividades compõem o histórico do tratamento de um incidente. A seguir, são apresentados dois exemplos de sequências de atividades que relatam como diferentes incidentes foram tratados:

Incidente #1:
Open → Assignment → Communication with customer → Resolved → Closed

Incidente #2:
Open → Assignment → Incident reproduction → Analysis/Research → Caused By CI → Closed

Critérios de avaliação



A cada etapa, serão apresentadas as 39 atividades deste processo e um grupo de atividades sugerido pelo algoritmo será destacado do conjunto para sua avaliação.

Você deve avaliar:

1) Se o grupo sugerido contém atividades que possuem um objetivo em comum;

2) Se existe alguma outra atividade que deveria pertencer ao grupo e não foi incluída;

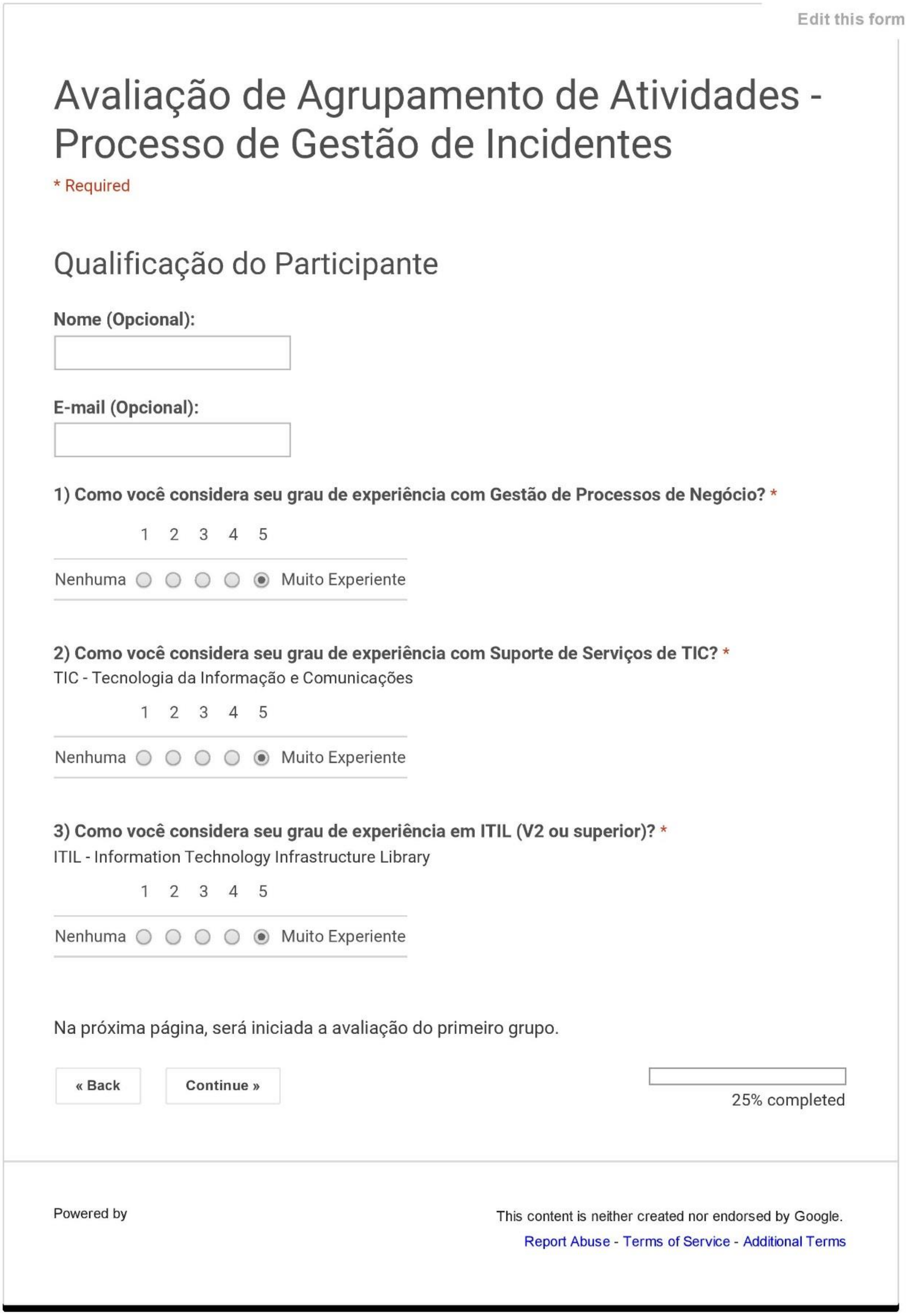
3) Se existe alguma atividade no grupo que deveria ser removida por não ter relação com as demais integrantes do grupo.

Não se preocupe com questões relacionadas à sequência de execução das atividades. O foco desta avaliação está no agrupamento das atividades com relação ao seu significado.

A seguir será apresentado o formulário de qualificação que nos auxiliará a definir o perfil dos respondentes.

This content is neither created nor endorsed by Google.

Report Abuse - Terms of Service - Additional Terms



Edit this form

Avaliação de Agrupamento de Atividades - Processo de Gestão de Incidentes

* Required

Qualificação do Participante

Nome (Opcional):

E-mail (Opcional):

1) Como você considera seu grau de experiência com Gestão de Processos de Negócio? *

	1	2	3	4	5	
Nenhuma	○	○	○	○	●	Muito Experiente

2) Como você considera seu grau de experiência com Suporte de Serviços de TIC? *

TIC - Tecnologia da Informação e Comunicações

	1	2	3	4	5	
Nenhuma	○	○	○	○	●	Muito Experiente

3) Como você considera seu grau de experiência em ITIL (V2 ou superior)? *

ITIL - Information Technology Infrastructure Library

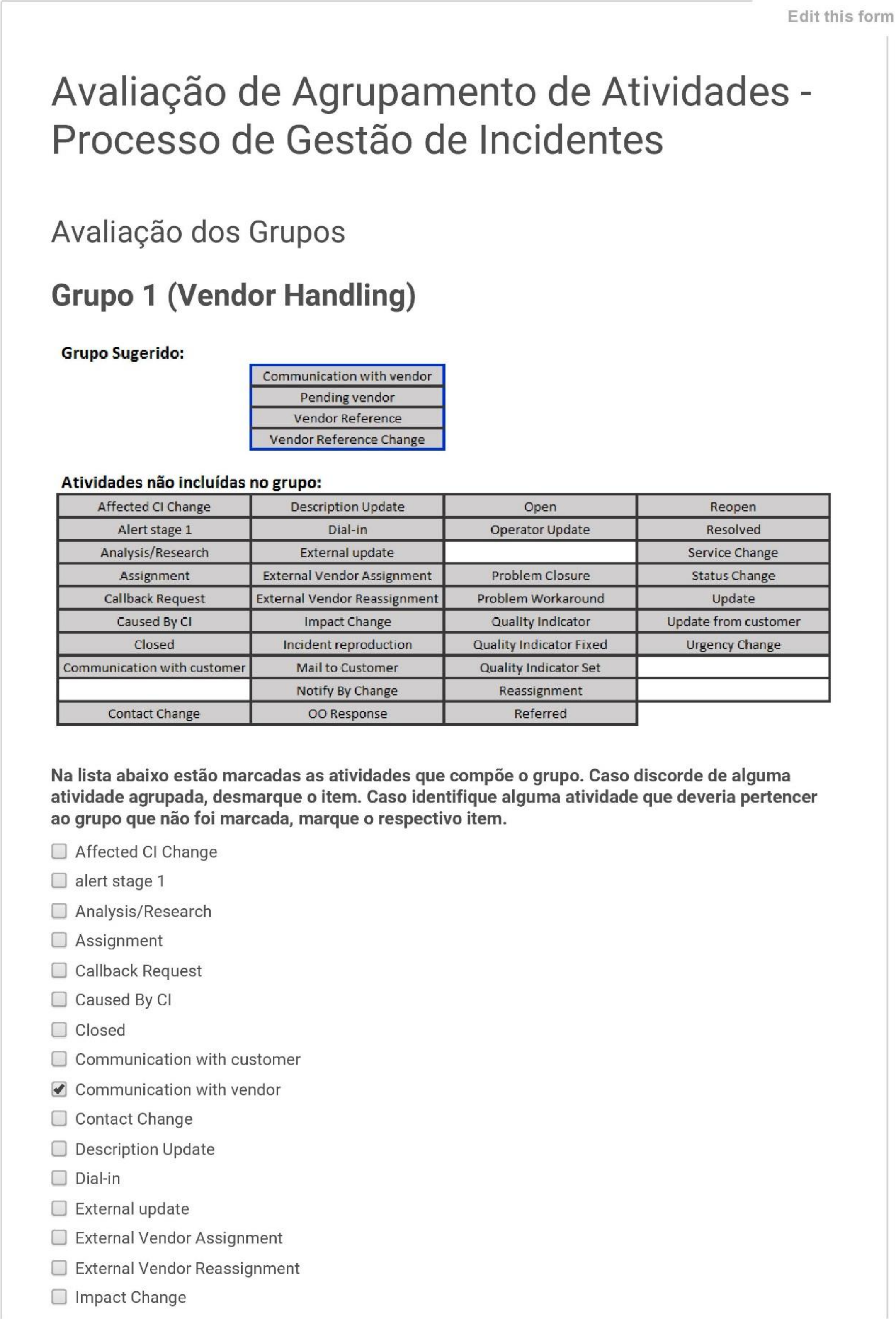
	1	2	3	4	5	
Nenhuma	○	○	○	○	●	Muito Experiente

Na próxima página, será iniciada a avaliação do primeiro grupo.

Powered by

This content is neither created nor endorsed by Google.

Report Abuse - Terms of Service - Additional Terms



Edit this form

Avaliação de Agrupamento de Atividades - Processo de Gestão de Incidentes

Avaliação dos Grupos

Grupo 1 (Vendor Handling)

Grupo Sugerido:

Communication with vendor
Pending vendor
Vendor Reference
Vendor Reference Change

Atividades não incluídas no grupo:

Affected CI Change	Description Update	Open	Reopen
Alert stage 1	Dial-in	Operator Update	Resolved
Analysis/Research	External update		Service Change
Assignment	External Vendor Assignment	Problem Closure	Status Change
Callback Request	External Vendor Reassignment	Problem Workaround	Update
Caused By CI	Impact Change	Quality Indicator	Update from customer
Closed	Incident reproduction	Quality Indicator Fixed	Urgency Change
Communication with customer	Mail to Customer	Quality Indicator Set	
	Notify By Change	Reassignment	
Contact Change	OO Response	Referred	

Na lista abaixo estão marcadas as atividades que compõe o grupo. Caso discorde de alguma atividade agrupada, desmarque o item. Caso identifique alguma atividade que deveria pertencer ao grupo que não foi marcada, marque o respectivo item.

☐ Affected CI Change

☐ alert stage 1

☐ Analysis/Research

☐ Assignment

☐ Callback Request

☐ Caused By CI

☐ Closed

☐ Communication with customer

☑ Communication with vendor

☐ Contact Change

☐ Description Update

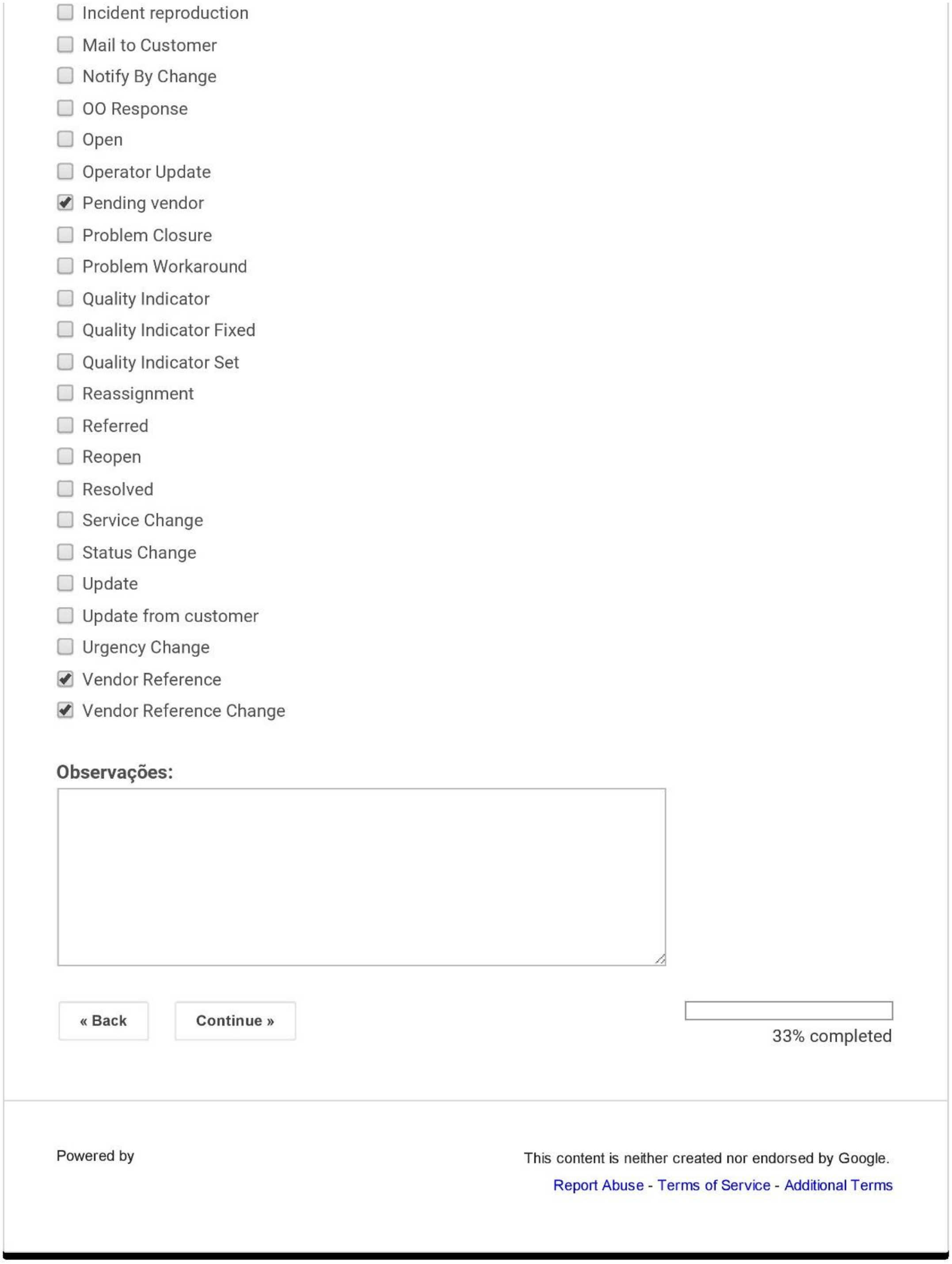
☐ Dial-in

☐ External update

☐ External Vendor Assignment

☐ External Vendor Reassignment

☐ Impact Change



□ Incident reproduction

□ Mail to Customer

□ Notify By Change

□ OO Response

□ Open

□ Operator Update

☑ Pending vendor

□ Problem Closure

□ Problem Workaround

□ Quality Indicator

□ Quality Indicator Fixed

□ Quality Indicator Set

□ Reassignment

□ Referred

□ Reopen

□ Resolved

□ Service Change

□ Status Change

□ Update

□ Update from customer

□ Urgency Change

☑ Vendor Reference

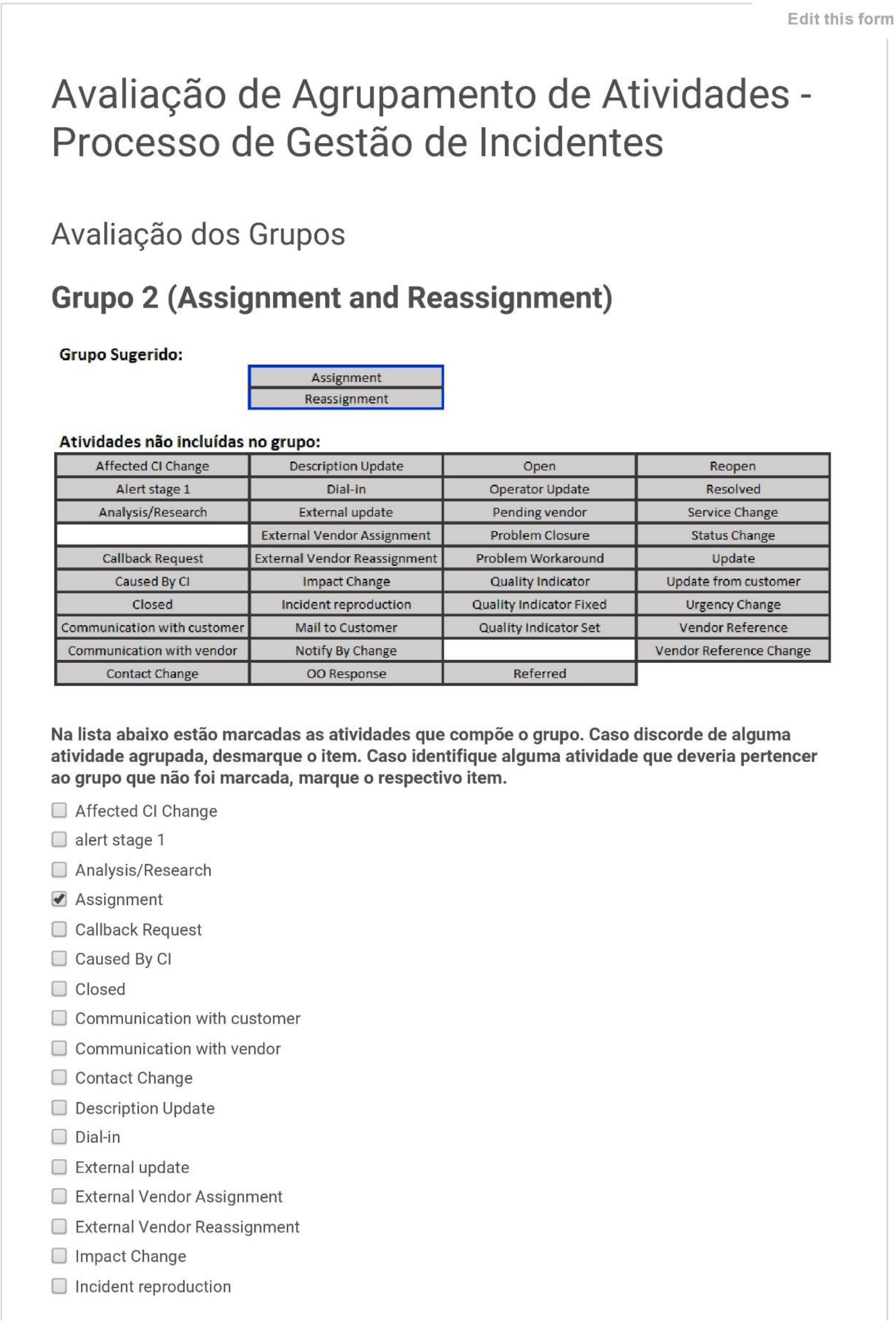
☑ Vendor Reference Change

Observações:

33% completed

Powered by

This content is neither created nor endorsed by Google.



Edit this form

Avaliação de Agrupamento de Atividades - Processo de Gestão de Incidentes

Avaliação dos Grupos

Grupo 2 (Assignment and Reassignment)

Grupo Sugerido:

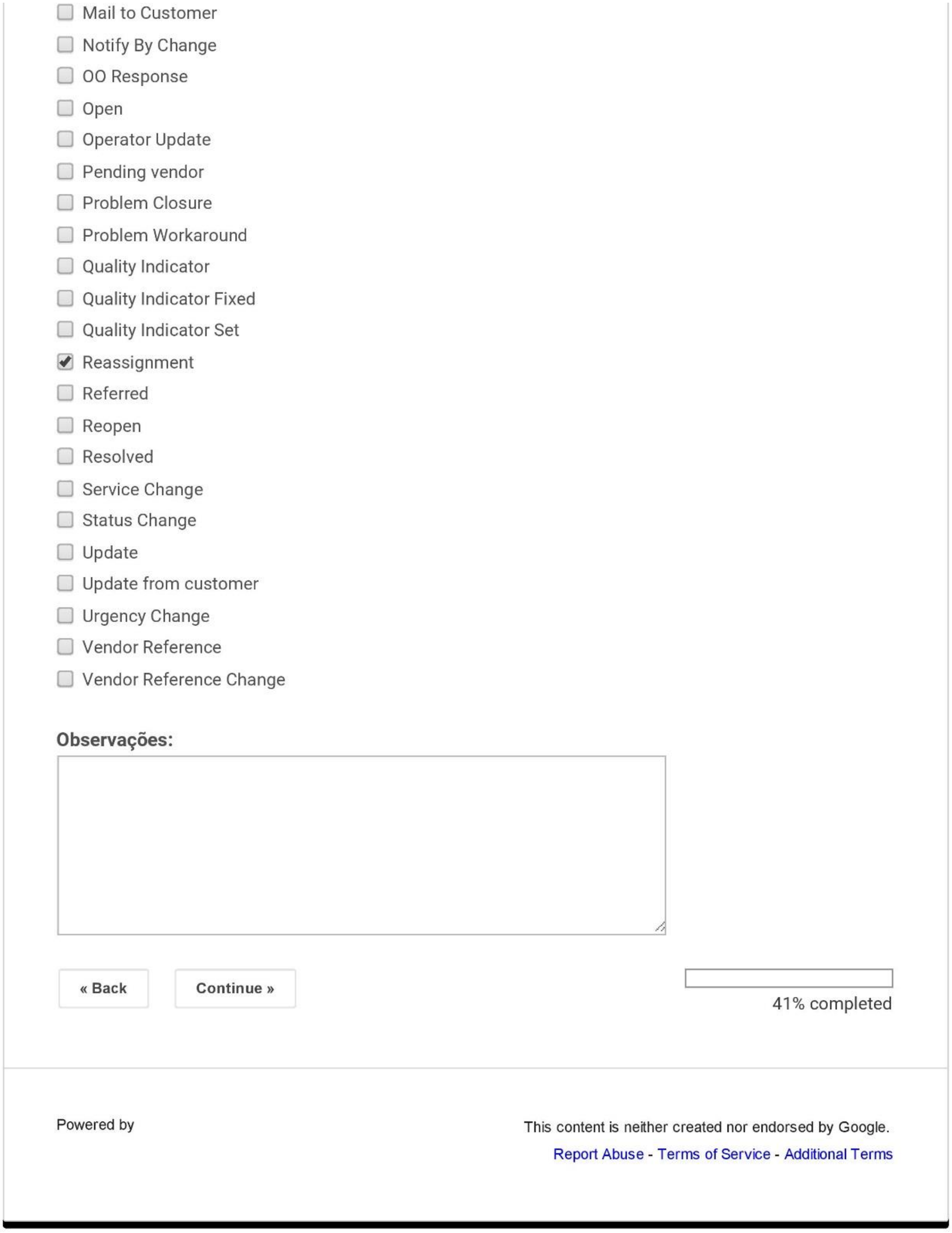
Assignment
Reassignment

Atividades não incluídas no grupo:

Affected CI Change	Description Update	Open	Reopen
Alert stage 1	Dial-in	Operator Update	Resolved
Analysis/Research	External update	Pending vendor	Service Change
	External Vendor Assignment	Problem Closure	Status Change
Callback Request	External Vendor Reassignment	Problem Workaround	Update
Caused By CI	Impact Change	Quality Indicator	Update from customer
Closed	Incident reproduction	Quality Indicator Fixed	Urgency Change
Communication with customer	Mail to Customer	Quality Indicator Set	Vendor Reference
Communication with vendor	Notify By Change		Vendor Reference Change
Contact Change	OO Response	Referred	

Na lista abaixo estão marcadas as atividades que compõe o grupo. Caso discorde de alguma atividade agrupada, desmarque o item. Caso identifique alguma atividade que deveria pertencer ao grupo que não foi marcada, marque o respectivo item.

- ☐ Affected CI Change
- ☐ alert stage 1
- ☐ Analysis/Research
- ☑ Assignment
- ☐ Callback Request
- ☐ Caused By CI
- ☐ Closed
- ☐ Communication with customer
- ☐ Communication with vendor
- ☐ Contact Change
- ☐ Description Update
- ☐ Dial-in
- ☐ External update
- ☐ External Vendor Assignment
- ☐ External Vendor Reassignment
- ☐ Impact Change
- ☐ Incident reproduction



□ Mail to Customer

□ Notify By Change

□ OO Response

□ Open

□ Operator Update

□ Pending vendor

□ Problem Closure

□ Problem Workaround

□ Quality Indicator

□ Quality Indicator Fixed

□ Quality Indicator Set

☑ Reassignment

□ Referred

□ Reopen

□ Resolved

□ Service Change

□ Status Change

□ Update

□ Update from customer

□ Urgency Change

□ Vendor Reference

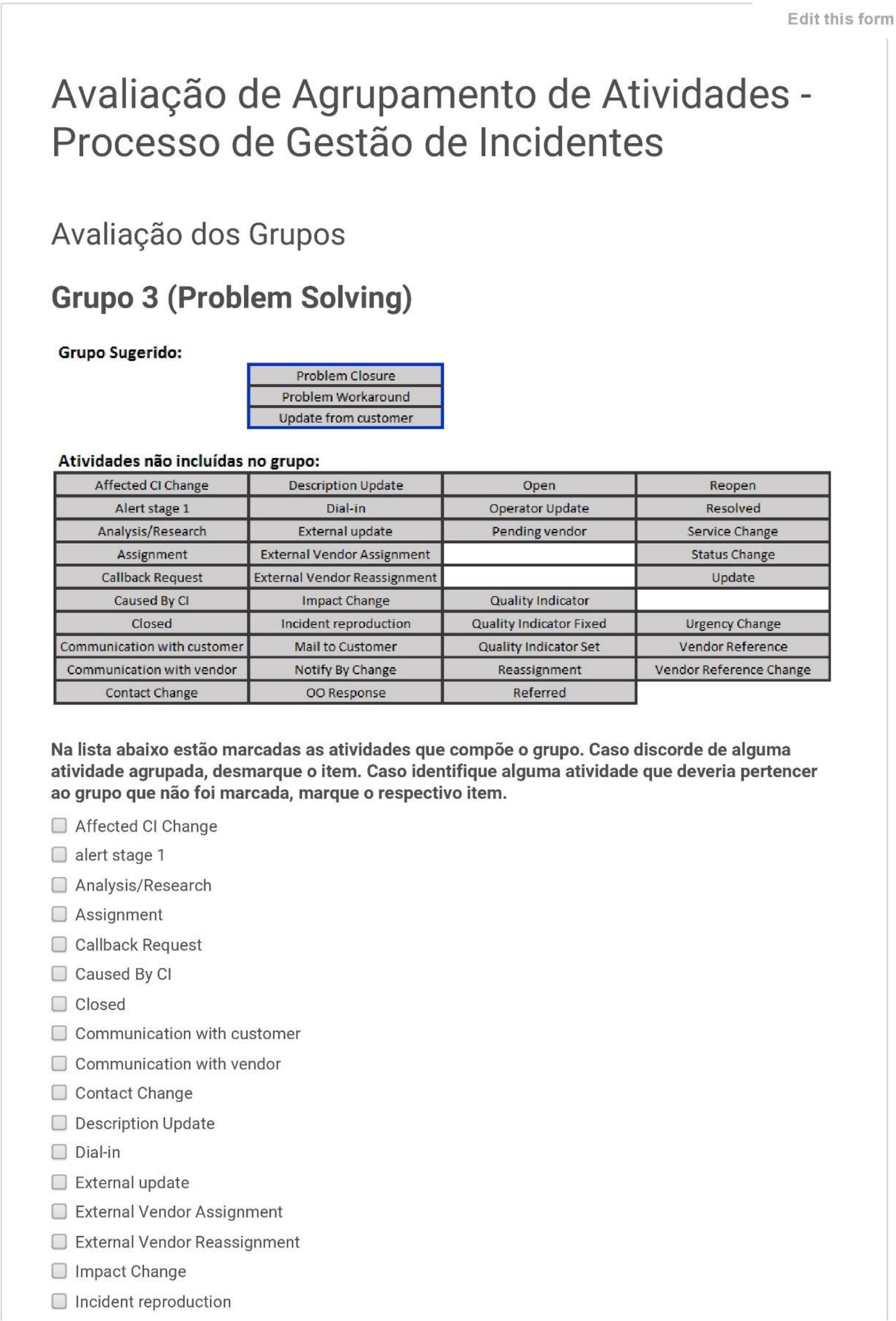
□ Vendor Reference Change

Observações:

Powered by

This content is neither created nor endorsed by Google.

Report Abuse - Terms of Service - Additional Terms



Avaliação de Agrupamento de Atividades - Processo de Gestão de Incidentes

Avaliação dos Grupos

Grupo 3 (Problem Solving)

Grupo Sugerido:

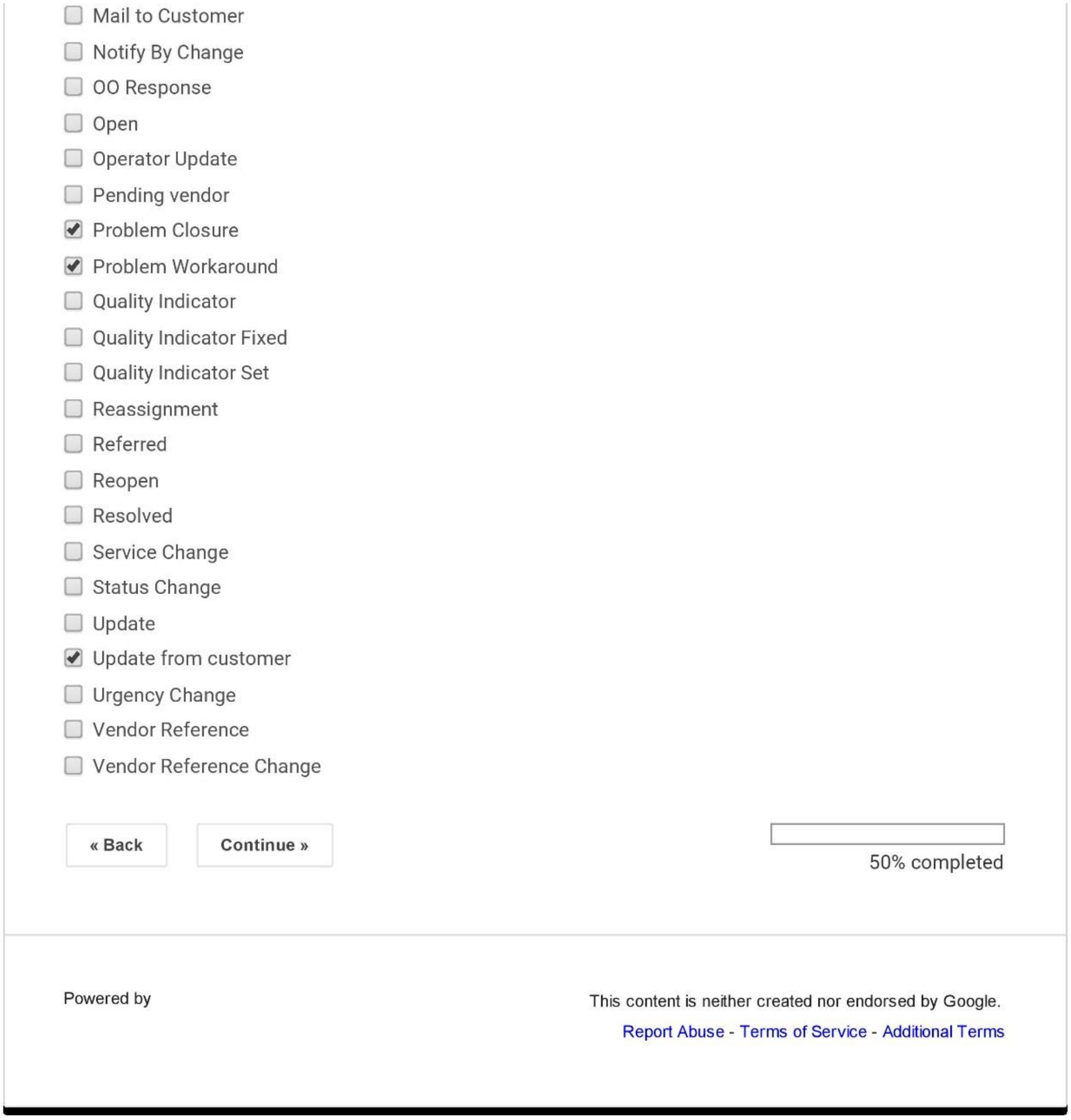
Problem Closure
Problem Workaround
Update from customer

Atividades não incluídas no grupo:

Affected CI Change	Description Update	Open	Reopen
Alert stage 1	Dial-in	Operator Update	Resolved
Analysis/Research	External update	Pending vendor	Service Change
Assignment	External Vendor Assignment		Status Change
Callback Request	External Vendor Reassignment		Update
Caused By CI	Impact Change	Quality Indicator	
Closed	Incident reproduction	Quality Indicator Fixed	Urgency Change
Communication with customer	Mail to Customer	Quality Indicator Set	Vendor Reference
Communication with vendor	Notify By Change	Reassignment	Vendor Reference Change
Contact Change	OO Response	Referred	

Na lista abaixo estão marcadas as atividades que compõe o grupo. Caso discorde de alguma atividade agrupada, desmarque o item. Caso identifique alguma atividade que deveria pertencer ao grupo que não foi marcada, marque o respectivo item.

- □ Affected CI Change
- □ alert stage 1
- □ Analysis/Research
- □ Assignment
- □ Callback Request
- □ Caused By CI
- □ Closed
- □ Communication with customer
- □ Communication with vendor
- □ Contact Change
- □ Description Update
- □ Dial-in
- □ External update
- □ External Vendor Assignment
- □ External Vendor Reassignment
- □ Impact Change
- □ Incident reproduction

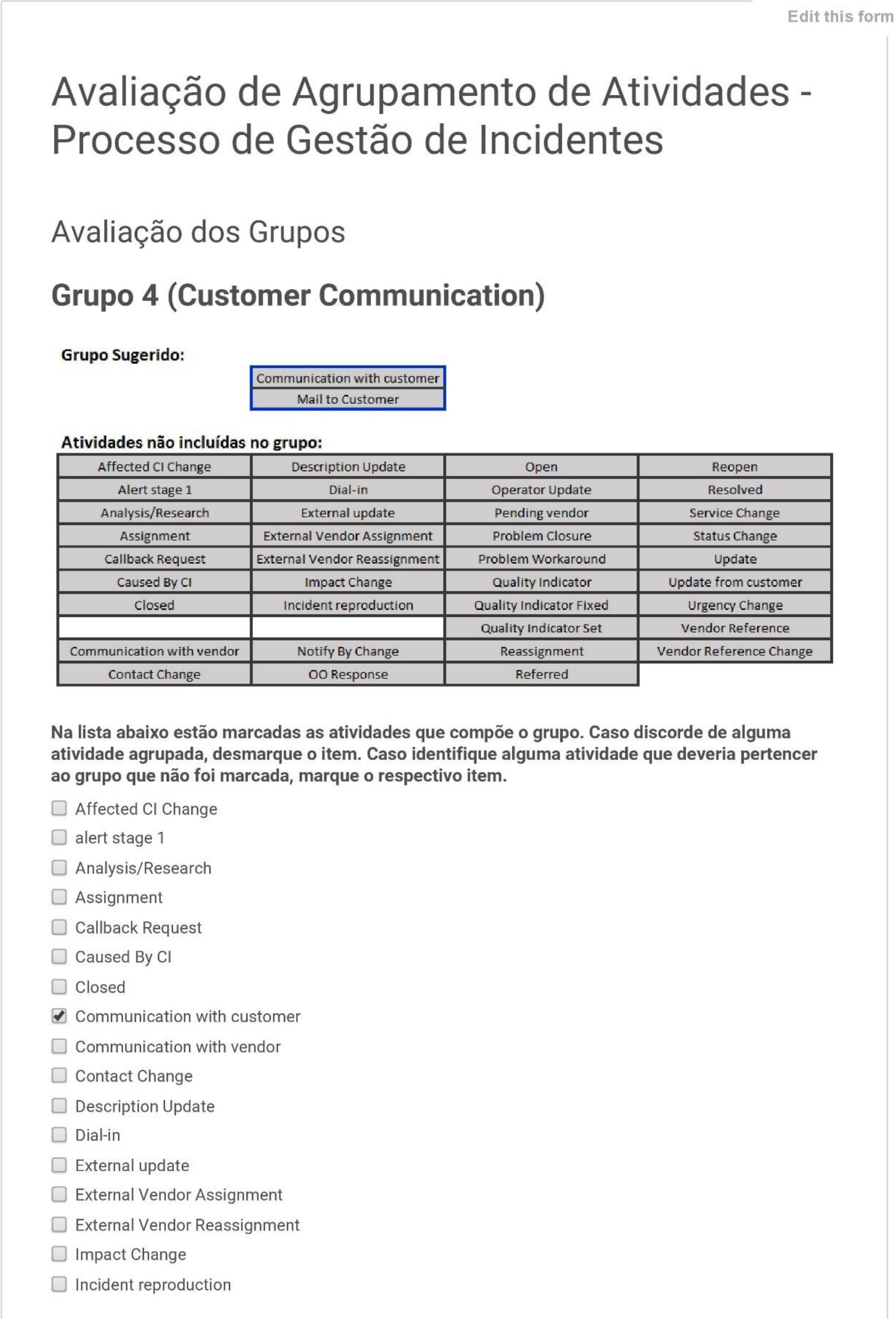


- ☐ Mail to Customer
- ☐ Notify By Change
- ☐ OO Response
- ☐ Open
- ☐ Operator Update
- ☐ Pending vendor
- ☑ Problem Closure
- ☑ Problem Workaround
- ☐ Quality Indicator
- ☐ Quality Indicator Fixed
- ☐ Quality Indicator Set
- ☐ Reassignment
- ☐ Referred
- ☐ Reopen
- ☐ Resolved
- ☐ Service Change
- ☐ Status Change
- ☐ Update
- ☑ Update from customer
- ☐ Urgency Change
- ☐ Vendor Reference
- ☐ Vendor Reference Change

Powered by

This content is neither created nor endorsed by Google.

Report Abuse - Terms of Service - Additional Terms



Edit this form

Avaliação de Agrupamento de Atividades - Processo de Gestão de Incidentes

Avaliação dos Grupos

Grupo 4 (Customer Communication)

Grupo Sugerido:

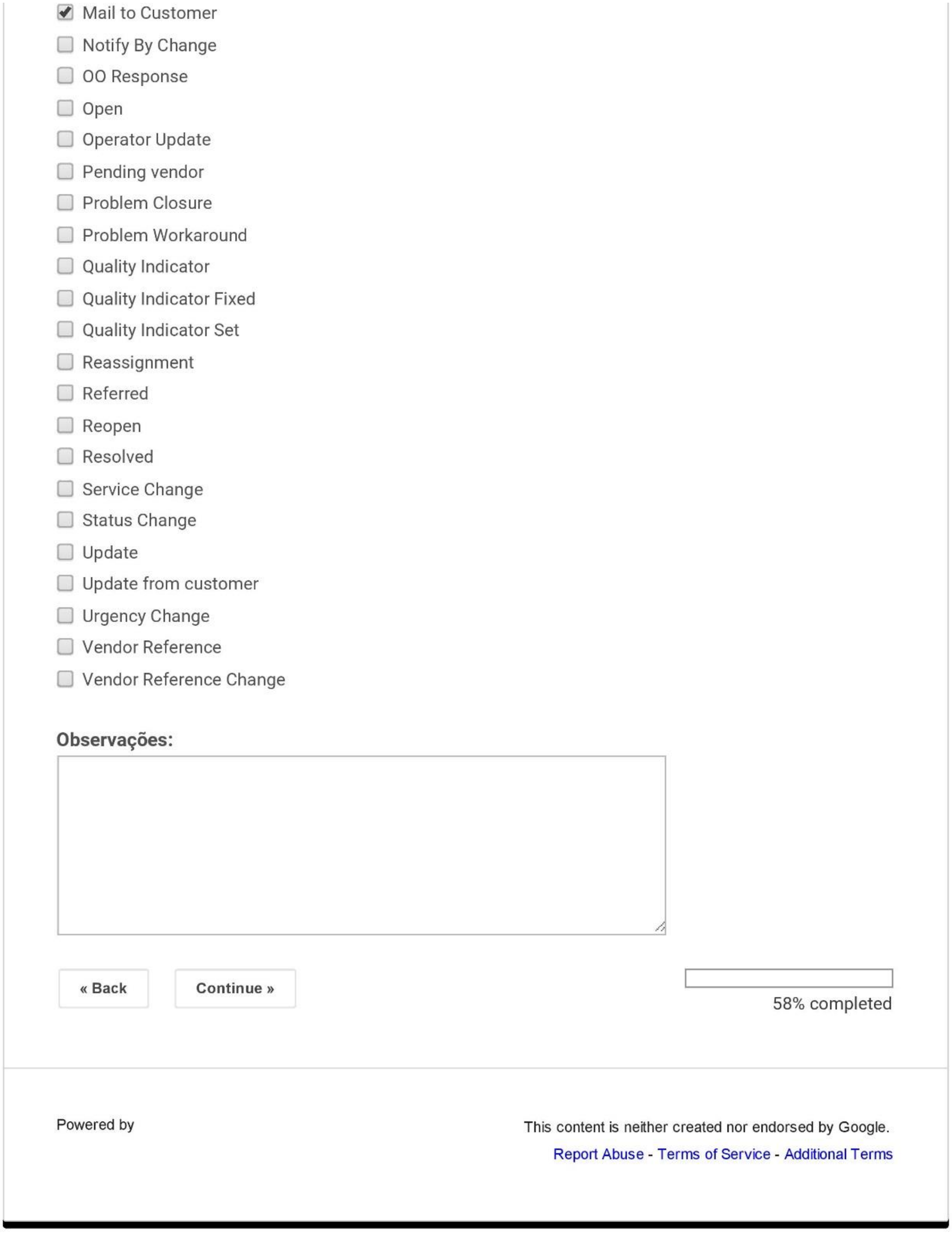
Communication with customer
Mail to Customer

Atividades não incluídas no grupo:

Affected CI Change	Description Update	Open	Reopen
Alert stage 1	Dial-in	Operator Update	Resolved
Analysis/Research	External update	Pending vendor	Service Change
Assignment	External Vendor Assignment	Problem Closure	Status Change
Callback Request	External Vendor Reassignment	Problem Workaround	Update
Caused By CI	Impact Change	Quality Indicator	Update from customer
Closed	Incident reproduction	Quality Indicator Fixed	Urgency Change
		Quality Indicator Set	Vendor Reference
Communication with vendor	Notify By Change	Reassignment	Vendor Reference Change
Contact Change	OO Response	Referred	

Na lista abaixo estão marcadas as atividades que compõe o grupo. Caso discorde de alguma atividade agrupada, desmarque o item. Caso identifique alguma atividade que deveria pertencer ao grupo que não foi marcada, marque o respectivo item.

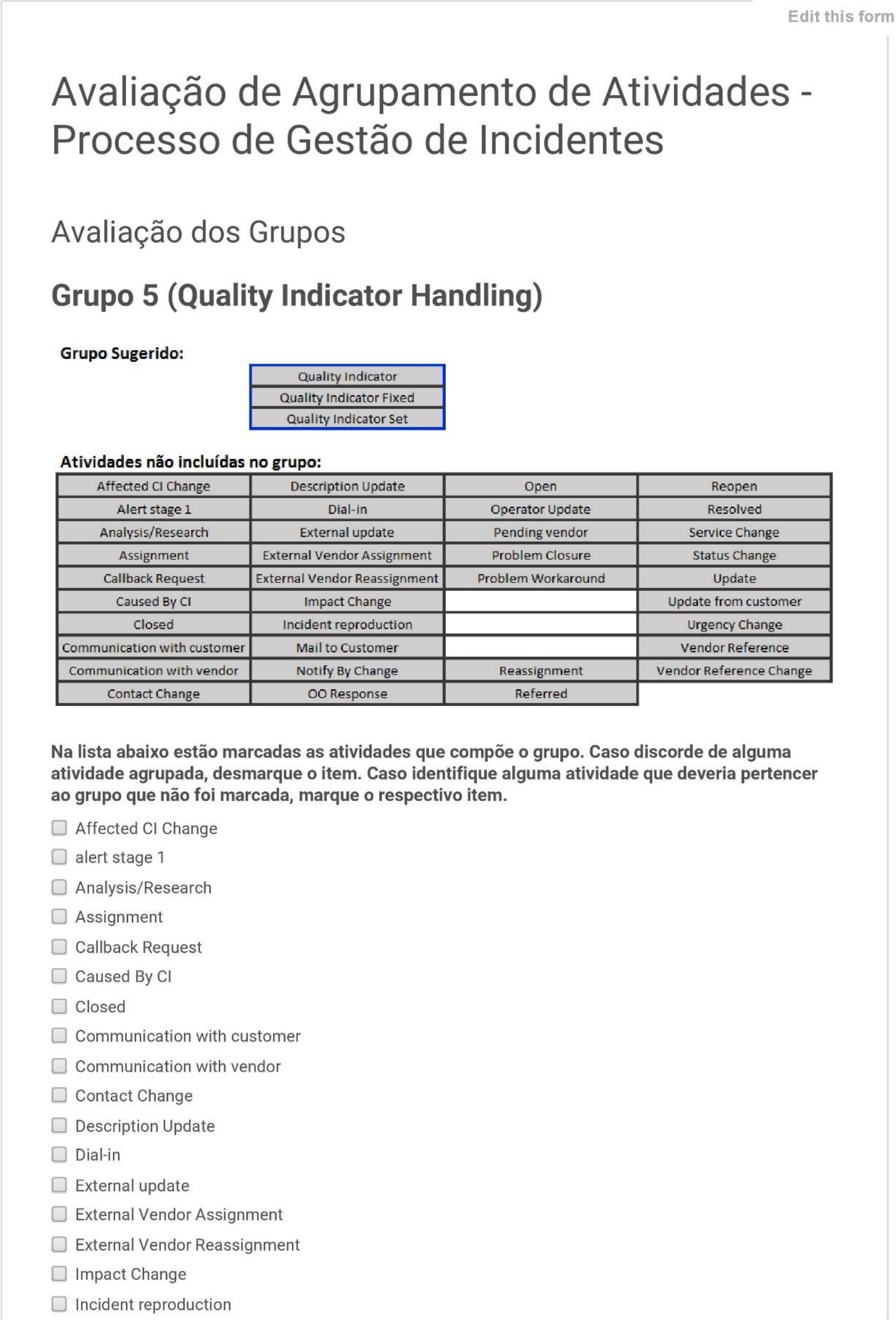
- □ Affected CI Change
- □ alert stage 1
- □ Analysis/Research
- □ Assignment
- □ Callback Request
- □ Caused By CI
- □ Closed
- ☑ Communication with customer
- □ Communication with vendor
- □ Contact Change
- □ Description Update
- □ Dial-in
- □ External update
- □ External Vendor Assignment
- □ External Vendor Reassignment
- □ Impact Change
- □ Incident reproduction



☑ Mail to Customer
□ Notify By Change
□ OO Response
□ Open
□ Operator Update
□ Pending vendor
□ Problem Closure
□ Problem Workaround
□ Quality Indicator
□ Quality Indicator Fixed
□ Quality Indicator Set
□ Reassignment
□ Referred
□ Reopen
□ Resolved
□ Service Change
□ Status Change
□ Update
□ Update from customer
□ Urgency Change
□ Vendor Reference
□ Vendor Reference Change

Observações:

« Back
Continue »
58% completed
Powered by
This content is neither created nor endorsed by Google.
Report Abuse - Terms of Service - Additional Terms



Edit this form

Avaliação de Agrupamento de Atividades - Processo de Gestão de Incidentes

Avaliação dos Grupos

Grupo 5 (Quality Indicator Handling)

Grupo Sugerido:

Quality Indicator
Quality Indicator Fixed
Quality Indicator Set

Atividades não incluídas no grupo:

Affected CI Change	Description Update	Open	Reopen
Alert stage 1	Dial-in	Operator Update	Resolved
Analysis/Research	External update	Pending vendor	Service Change
Assignment	External Vendor Assignment	Problem Closure	Status Change
Callback Request	External Vendor Reassignment	Problem Workaround	Update
Caused By CI	Impact Change		Update from customer
Closed	Incident reproduction		Urgency Change
Communication with customer	Mail to Customer		Vendor Reference
Communication with vendor	Notify By Change	Reassignment	Vendor Reference Change
Contact Change	OO Response	Referred	

Na lista abaixo estão marcadas as atividades que compõe o grupo. Caso discorde de alguma atividade agrupada, desmarque o item. Caso identifique alguma atividade que deveria pertencer ao grupo que não foi marcada, marque o respectivo item.

□ Affected CI Change

□ alert stage 1

□ Analysis/Research

□ Assignment

□ Callback Request

□ Caused By CI

□ Closed

□ Communication with customer

□ Communication with vendor

□ Contact Change

□ Description Update

□ Dial-in

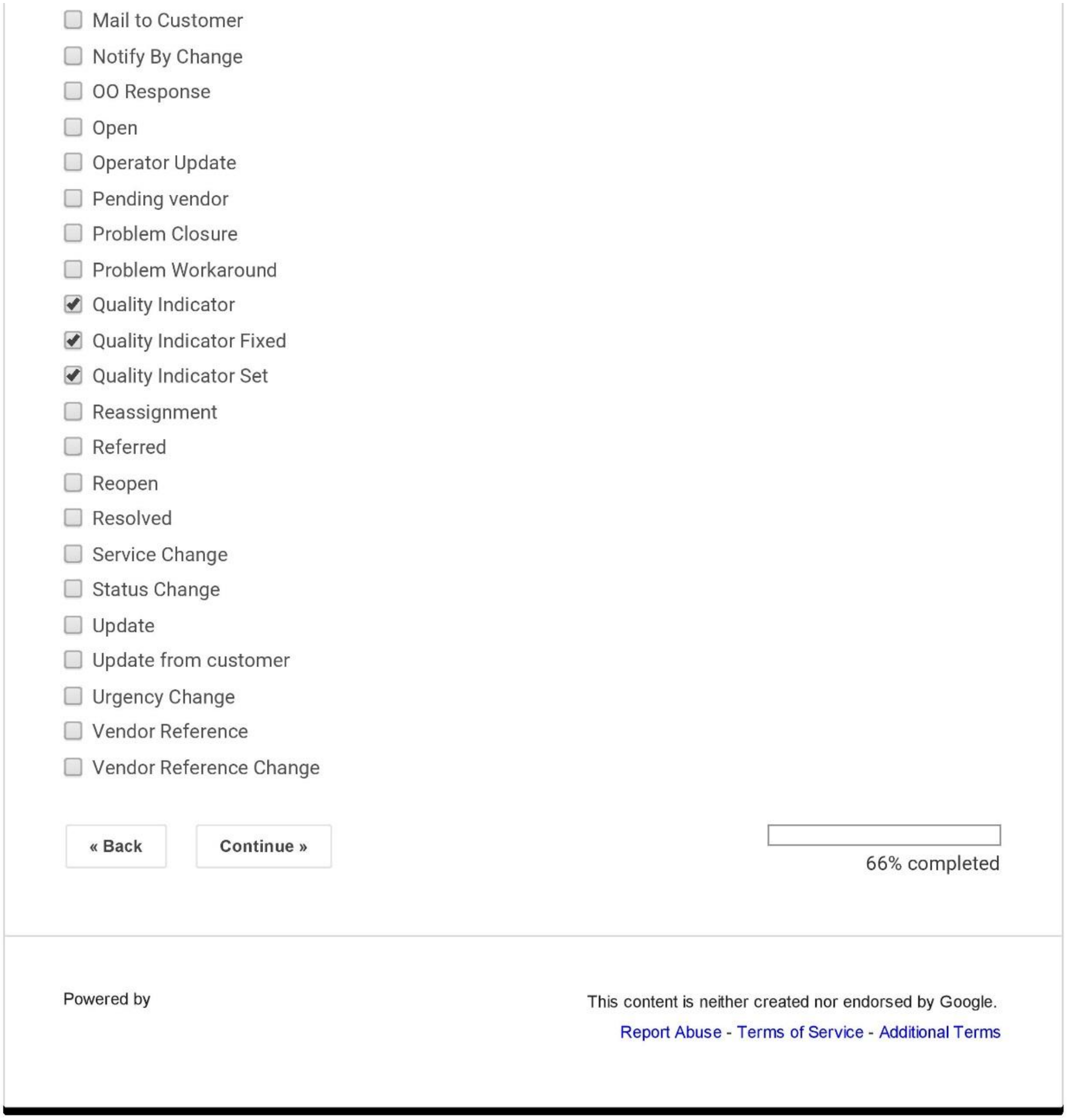
□ External update

□ External Vendor Assignment

□ External Vendor Reassignment

□ Impact Change

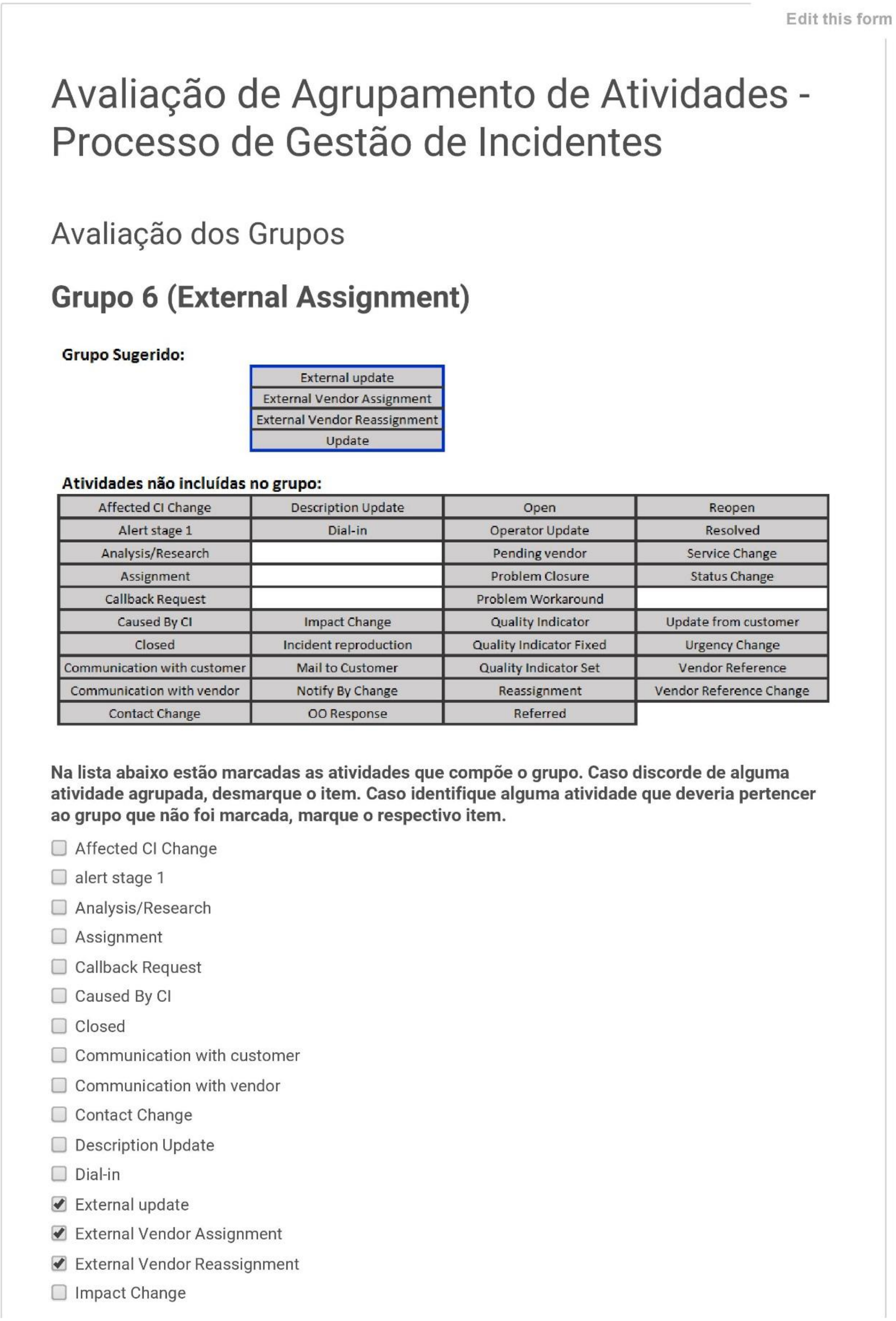
□ Incident reproduction



- □ Mail to Customer
- □ Notify By Change
- □ OO Response
- □ Open
- □ Operator Update
- □ Pending vendor
- □ Problem Closure
- □ Problem Workaround
- ☑ Quality Indicator
- ☑ Quality Indicator Fixed
- ☑ Quality Indicator Set
- □ Reassignment
- □ Referred
- □ Reopen
- □ Resolved
- □ Service Change
- □ Status Change
- □ Update
- □ Update from customer
- □ Urgency Change
- □ Vendor Reference
- □ Vendor Reference Change

Powered by

This content is neither created nor endorsed by Google.
Report Abuse - Terms of Service - Additional Terms



Edit this form

Avaliação de Agrupamento de Atividades - Processo de Gestão de Incidentes

Avaliação dos Grupos

Grupo 6 (External Assignment)

Grupo Sugerido:

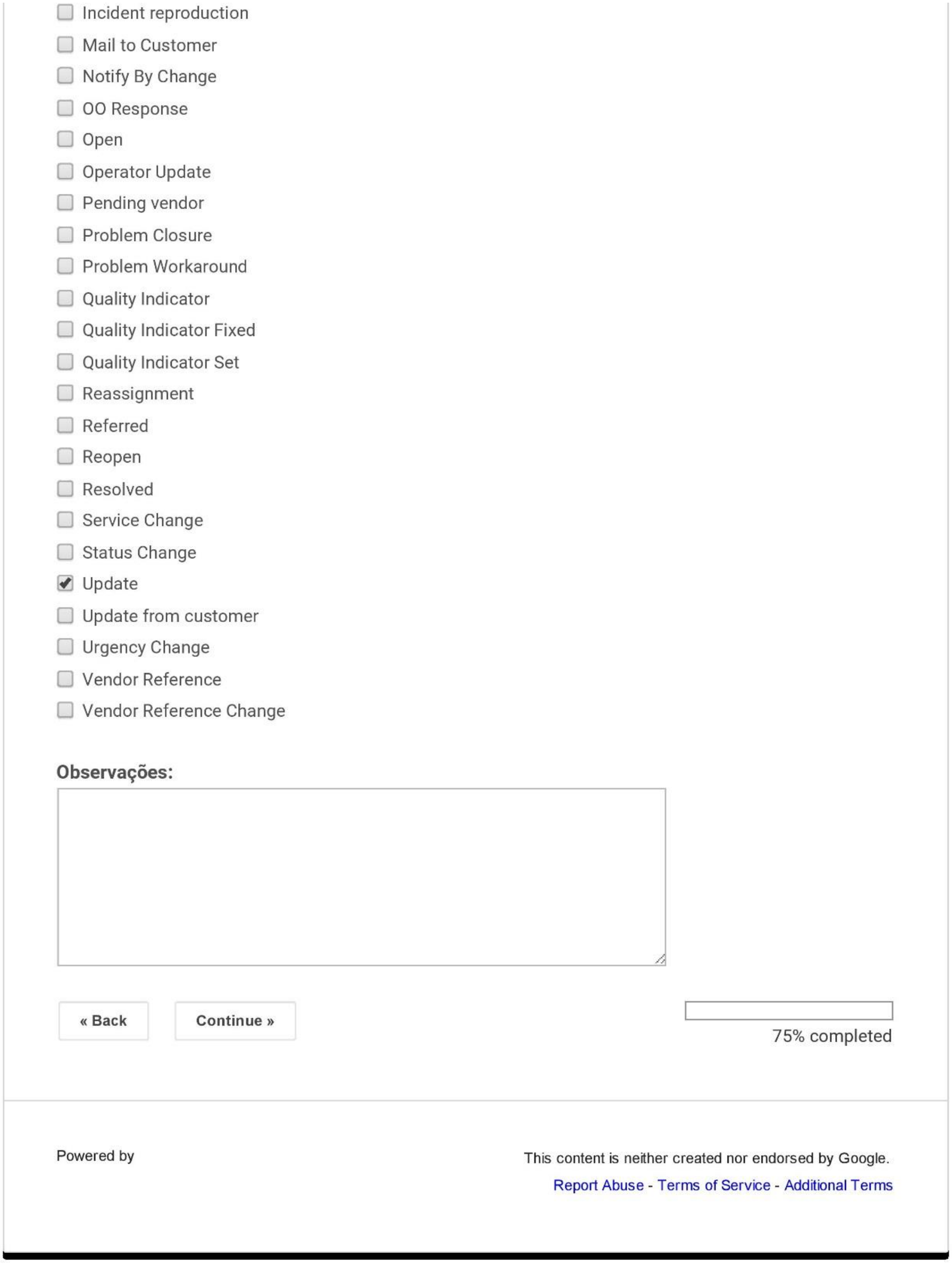
External update
External Vendor Assignment
External Vendor Reassignment
Update

Atividades não incluídas no grupo:

Affected CI Change	Description Update	Open	Reopen
Alert stage 1	Dial-in	Operator Update	Resolved
Analysis/Research		Pending vendor	Service Change
Assignment		Problem Closure	Status Change
Callback Request		Problem Workaround	
Caused By CI	Impact Change	Quality Indicator	Update from customer
Closed	Incident reproduction	Quality Indicator Fixed	Urgency Change
Communication with customer	Mail to Customer	Quality Indicator Set	Vendor Reference
Communication with vendor	Notify By Change	Reassignment	Vendor Reference Change
Contact Change	OO Response	Referred	

Na lista abaixo estão marcadas as atividades que compõe o grupo. Caso discorde de alguma atividade agrupada, desmarque o item. Caso identifique alguma atividade que deveria pertencer ao grupo que não foi marcada, marque o respectivo item.

- ☐ Affected CI Change
- ☐ alert stage 1
- ☐ Analysis/Research
- ☐ Assignment
- ☐ Callback Request
- ☐ Caused By CI
- ☐ Closed
- ☐ Communication with customer
- ☐ Communication with vendor
- ☐ Contact Change
- ☐ Description Update
- ☐ Dial-in
- ☑ External update
- ☑ External Vendor Assignment
- ☑ External Vendor Reassignment
- ☐ Impact Change

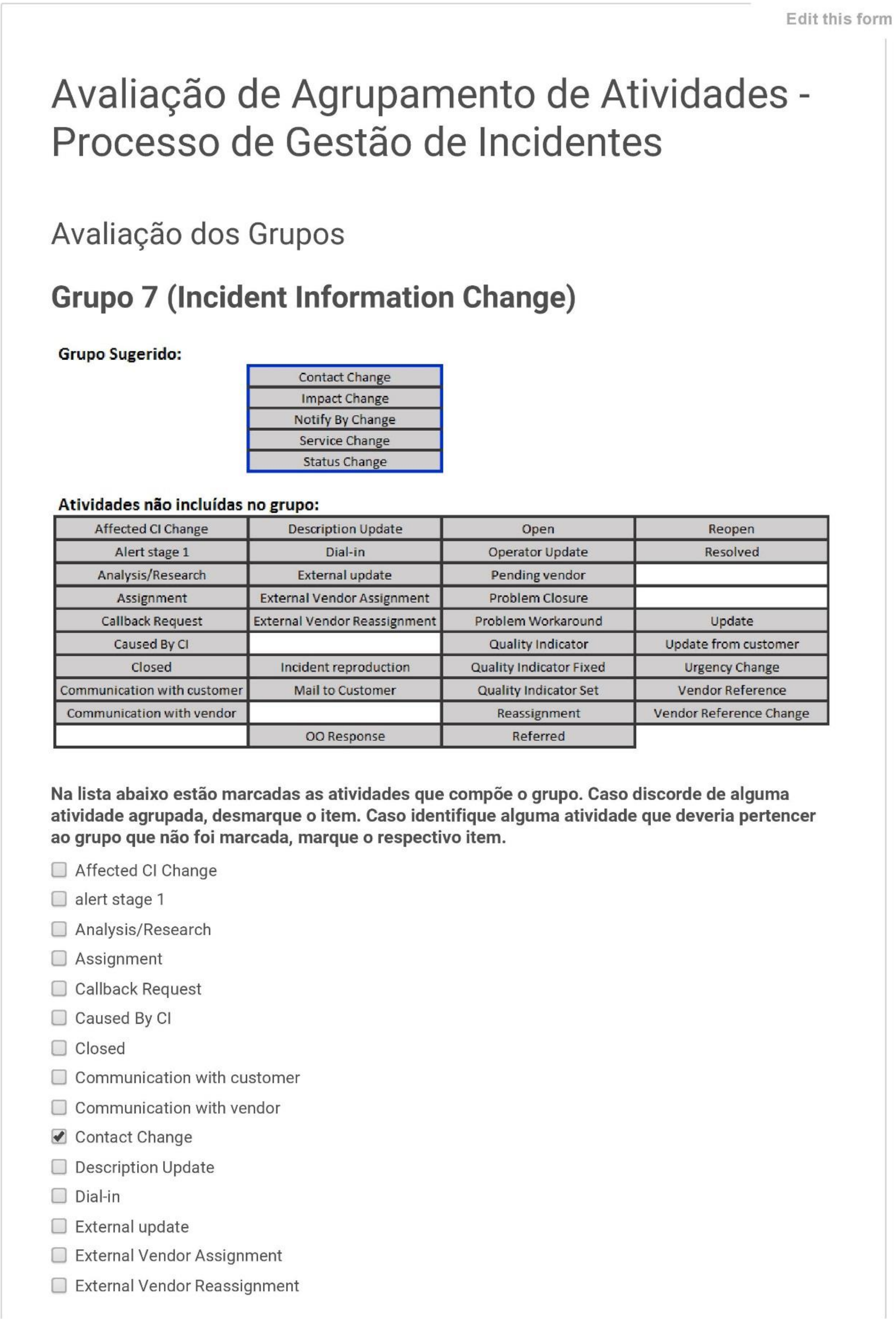


- ☐ Incident reproduction
- ☐ Mail to Customer
- ☐ Notify By Change
- ☐ OO Response
- ☐ Open
- ☐ Operator Update
- ☐ Pending vendor
- ☐ Problem Closure
- ☐ Problem Workaround
- ☐ Quality Indicator
- ☐ Quality Indicator Fixed
- ☐ Quality Indicator Set
- ☐ Reassignment
- ☐ Referred
- ☐ Reopen
- ☐ Resolved
- ☐ Service Change
- ☐ Status Change
- ☑ Update
- ☐ Update from customer
- ☐ Urgency Change
- ☐ Vendor Reference
- ☐ Vendor Reference Change

Observações:

Powered by

This content is neither created nor endorsed by Google.



Edit this form

Avaliação de Agrupamento de Atividades - Processo de Gestão de Incidentes

Avaliação dos Grupos

Grupo 7 (Incident Information Change)

Grupo Sugerido:

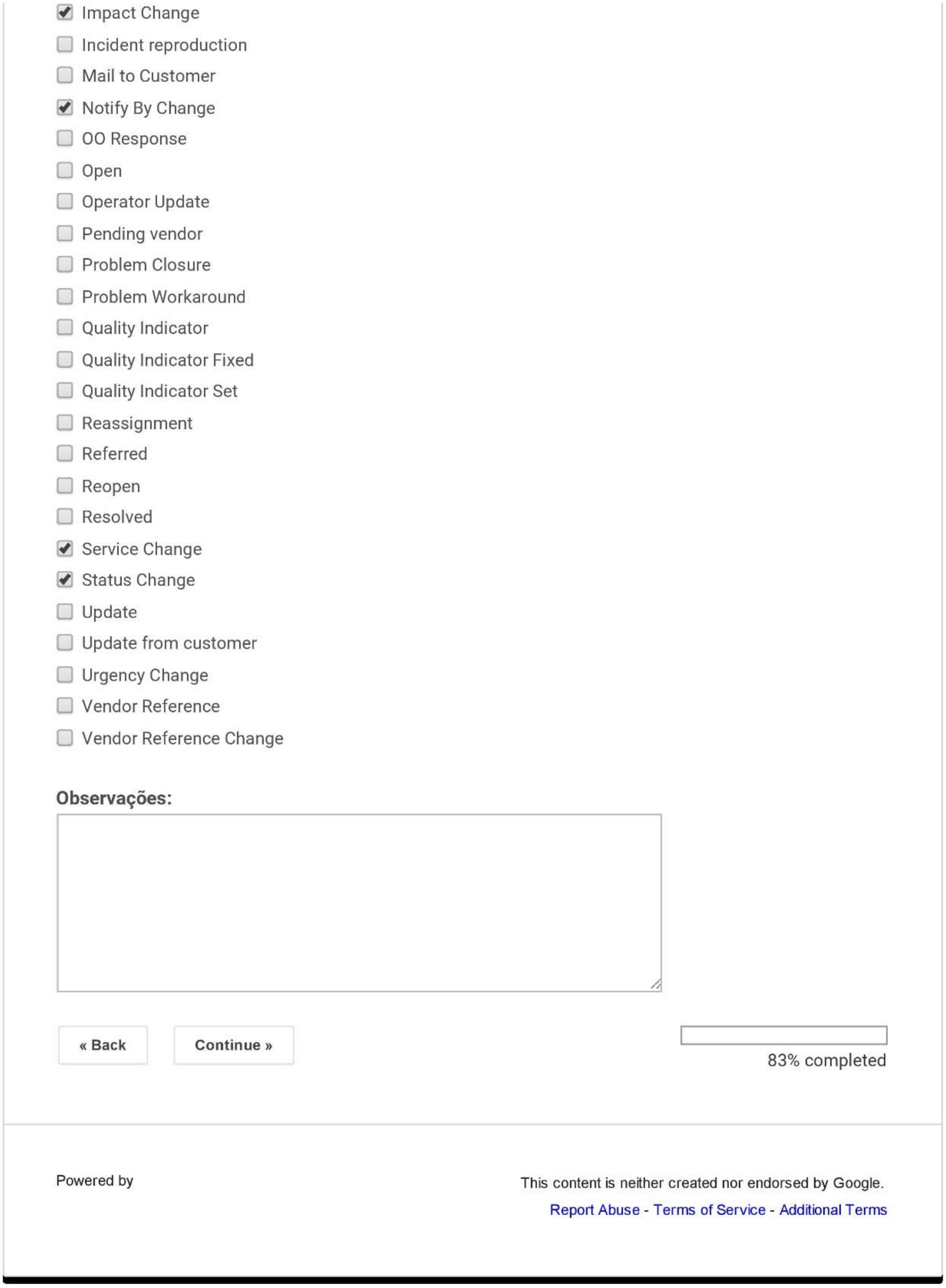
Contact Change
Impact Change
Notify By Change
Service Change
Status Change

Atividades não incluídas no grupo:

Affected CI Change	Description Update	Open	Reopen
Alert stage 1	Dial-in	Operator Update	Resolved
Analysis/Research	External update	Pending vendor	
Assignment	External Vendor Assignment	Problem Closure	
Callback Request	External Vendor Reassignment	Problem Workaround	Update
Caused By CI		Quality Indicator	Update from customer
Closed	Incident reproduction	Quality Indicator Fixed	Urgency Change
Communication with customer	Mail to Customer	Quality Indicator Set	Vendor Reference
Communication with vendor		Reassignment	Vendor Reference Change
	OO Response	Referred	

Na lista abaixo estão marcadas as atividades que compõe o grupo. Caso discorde de alguma atividade agrupada, desmarque o item. Caso identifique alguma atividade que deveria pertencer ao grupo que não foi marcada, marque o respectivo item.

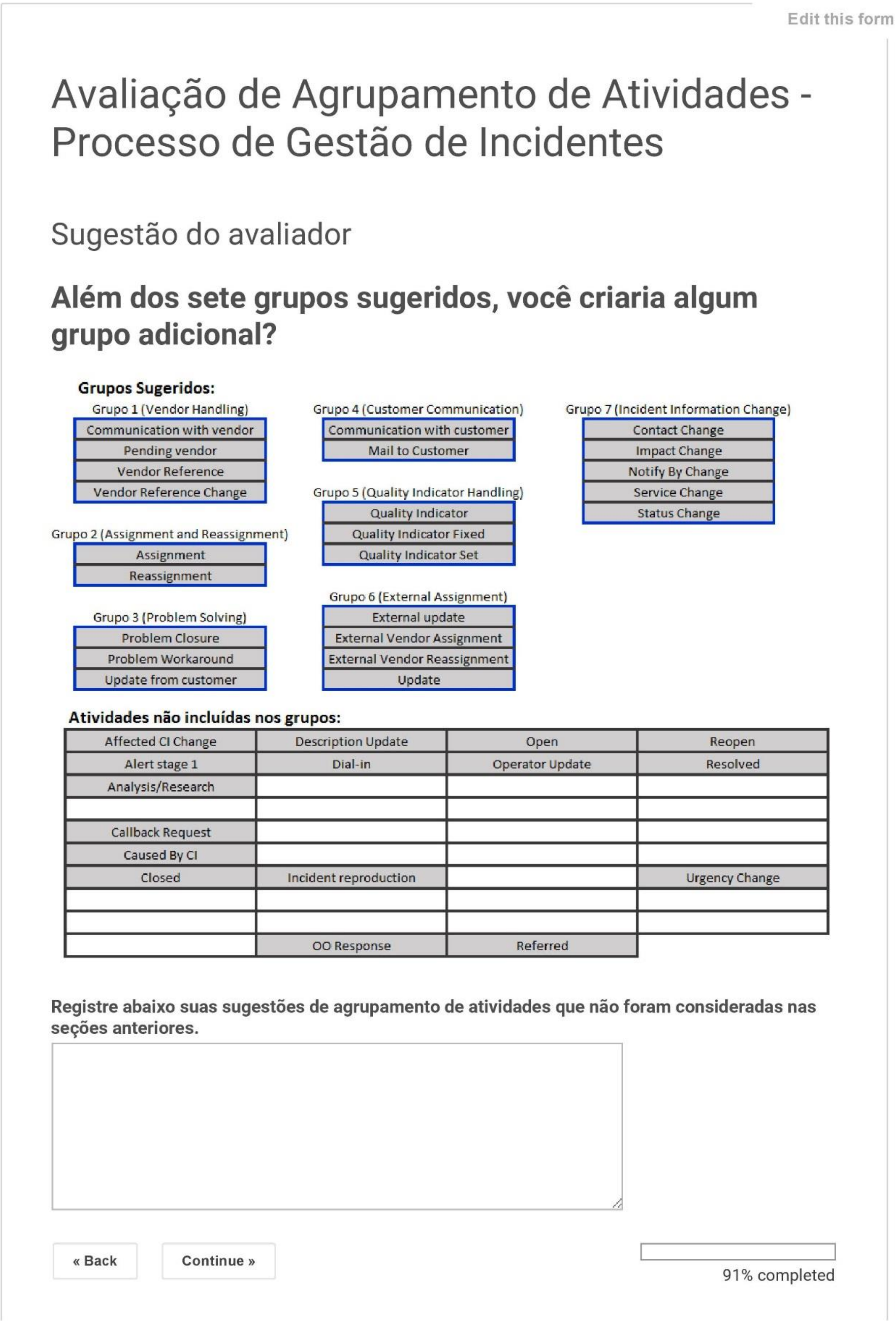
- □ Affected CI Change
- □ alert stage 1
- □ Analysis/Research
- □ Assignment
- □ Callback Request
- □ Caused By CI
- □ Closed
- □ Communication with customer
- □ Communication with vendor
- ☑ Contact Change
- □ Description Update
- □ Dial-in
- □ External update
- □ External Vendor Assignment
- □ External Vendor Reassignment



☑ Impact Change
□ Incident reproduction
□ Mail to Customer
☑ Notify By Change
□ OO Response
□ Open
□ Operator Update
□ Pending vendor
□ Problem Closure
□ Problem Workaround
□ Quality Indicator
□ Quality Indicator Fixed
□ Quality Indicator Set
□ Reassignment
□ Referred
□ Reopen
□ Resolved
☑ Service Change
☑ Status Change
□ Update
□ Update from customer
□ Urgency Change
□ Vendor Reference
□ Vendor Reference Change

Observações:

« Back
Continue »
83% completed
Powered by
This content is neither created nor endorsed by Google.
Report Abuse - Terms of Service - Additional Terms



Edit this form

Avaliação de Agrupamento de Atividades - Processo de Gestão de Incidentes

Sugestão do avaliador

Além dos sete grupos sugeridos, você criaria algum grupo adicional?

Grupos Sugeridos:

Grupo 1 (Vendor Handling)

- Communication with vendor
- Pending vendor
- Vendor Reference
- Vendor Reference Change

Grupo 2 (Assignment and Reassignment)

- Assignment
- Reassignment

Grupo 3 (Problem Solving)

- Problem Closure
- Problem Workaround
- Update from customer

Grupo 4 (Customer Communication)

- Communication with customer
- Mail to Customer

Grupo 5 (Quality Indicator Handling)

- Quality Indicator
- Quality Indicator Fixed
- Quality Indicator Set

Grupo 6 (External Assignment)

- External update
- External Vendor Assignment
- External Vendor Reassignment
- Update

Grupo 7 (Incident Information Change)

- Contact Change
- Impact Change
- Notify By Change
- Service Change
- Status Change

Atividades não incluídas nos grupos:

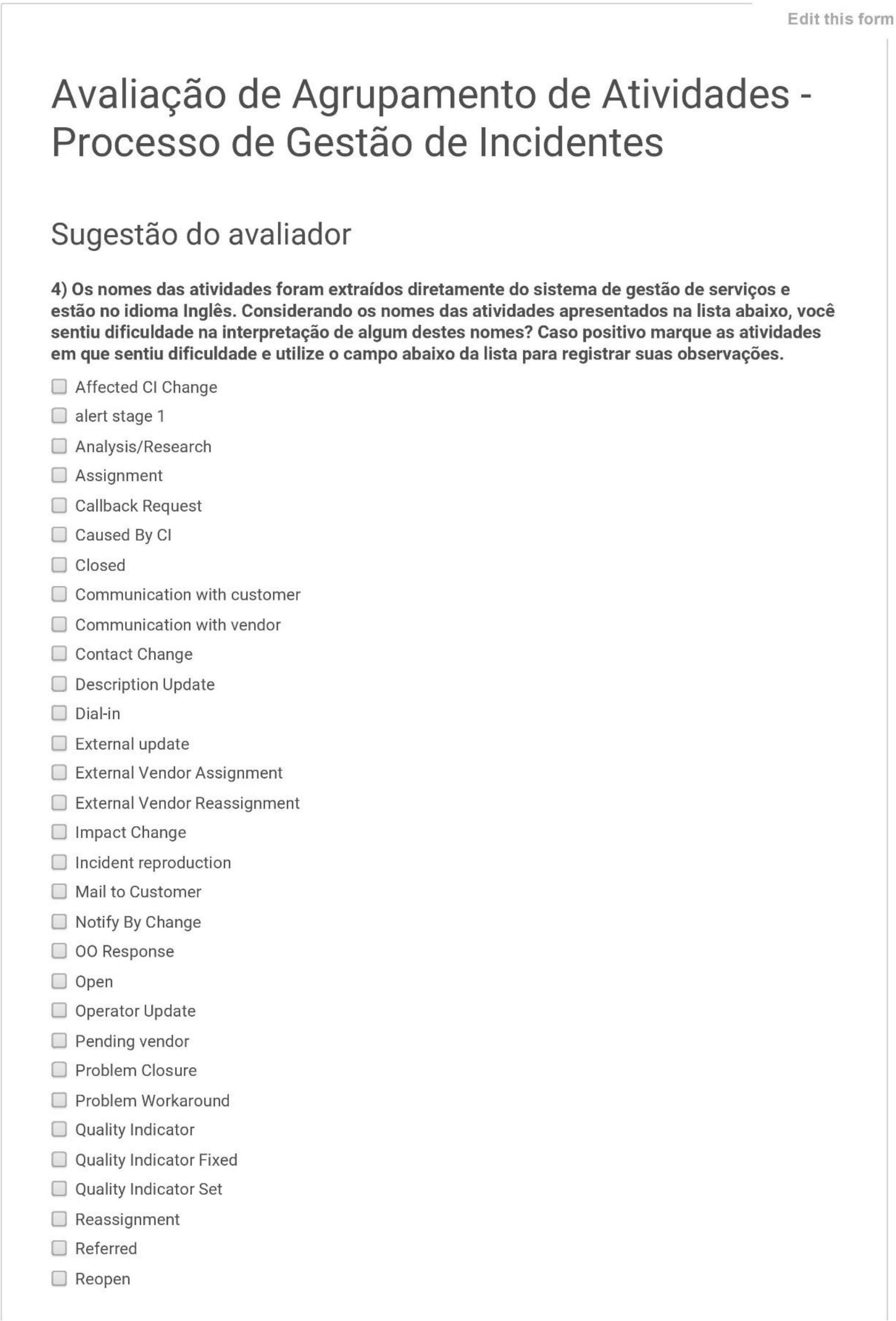
Affected CI Change	Description Update	Open	Reopen
Alert stage 1	Dial-in	Operator Update	Resolved
Analysis/Research			
Callback Request			
Caused By CI			
Closed	Incident reproduction		Urgency Change
	OO Response	Referred	

Registre abaixo suas sugestões de agrupamento de atividades que não foram consideradas nas seções anteriores.

« Back

Continue »

91% completed



Edit this form

Avaliação de Agrupamento de Atividades - Processo de Gestão de Incidentes

Sugestão do avaliador

4) Os nomes das atividades foram extraídos diretamente do sistema de gestão de serviços e estão no idioma Inglês. Considerando os nomes das atividades apresentados na lista abaixo, você sentiu dificuldade na interpretação de algum destes nomes? Caso positivo marque as atividades em que sentiu dificuldade e utilize o campo abaixo da lista para registrar suas observações.

□ Affected CI Change

□ alert stage 1

□ Analysis/Research

□ Assignment

□ Callback Request

□ Caused By CI

□ Closed

□ Communication with customer

□ Communication with vendor

□ Contact Change

□ Description Update

□ Dial-in

□ External update

□ External Vendor Assignment

□ External Vendor Reassignment

□ Impact Change

□ Incident reproduction

□ Mail to Customer

□ Notify By Change

□ OO Response

□ Open

□ Operator Update

□ Pending vendor

□ Problem Closure

□ Problem Workaround

□ Quality Indicator

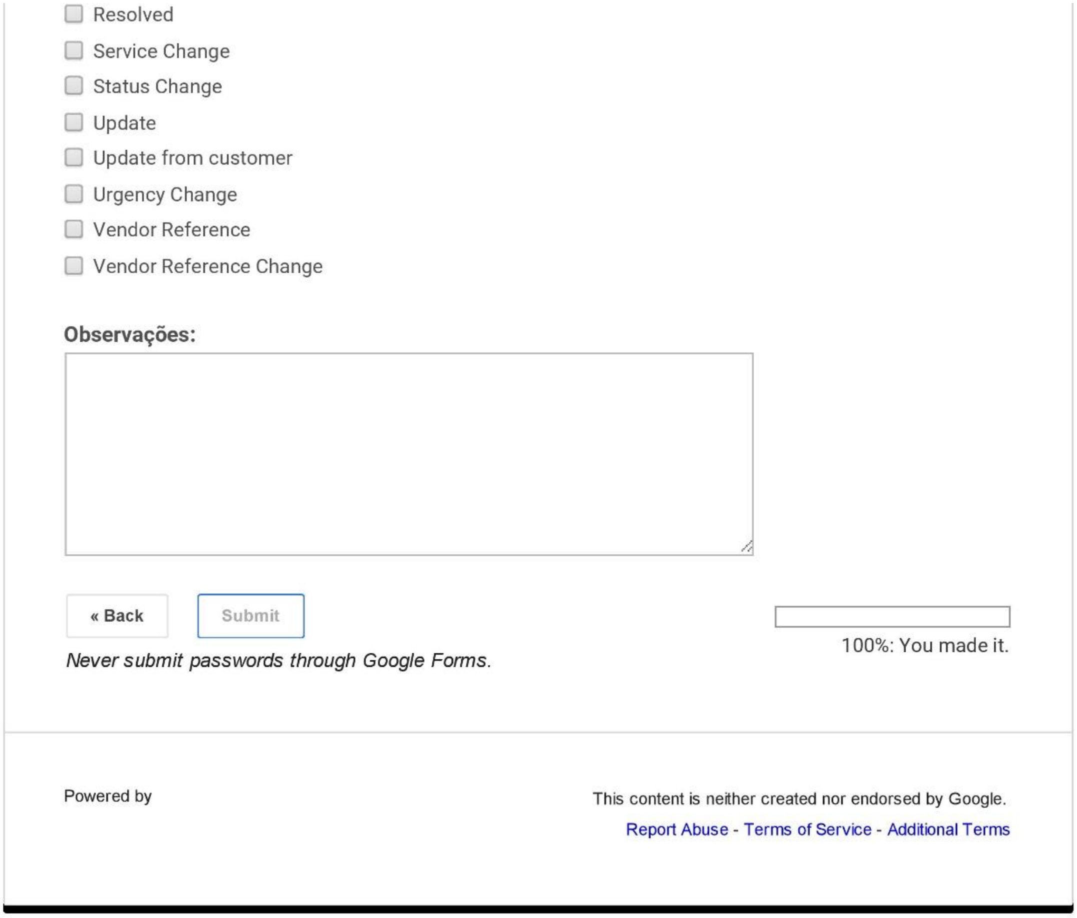
□ Quality Indicator Fixed

□ Quality Indicator Set

□ Reassignment

□ Referred

□ Reopen



□ Resolved

□ Service Change

□ Status Change

□ Update

□ Update from customer

□ Urgency Change

□ Vendor Reference

□ Vendor Reference Change

Observações:

Never submit passwords through Google Forms.

100%: You made it.

Powered by

This content is neither created nor endorsed by Google.

Report Abuse - Terms of Service - Additional Terms