UNIVERSITÄT KOBLENZ · LANDAU

IWVI

Survey of Business Process Modeling Recommender Systems

Masterarbeit

Zur Erlangung des Grades eines Master of Science im Studiengang Web Science

vorgelegt von

Abdullah Imad Abdullah Elkindy

[216100870]

Koblenz, im Juli 2019

Erstgutachter: Prof. Dr. Patrick Delfmann
(Institut für Wirtschafts- und Verwaltungsinformatik, FG Delfmann)

Zweitgutachter: M.Sc. Carl Corea
(Institut für Wirtschafts- und Verwaltungsinformatik, FG Delfmann)

Eidesstattliche Erklärung

Hiermit bestätige ich, dass die vorliegende Arbeit von mir selbständig verfasst wurde und ich keine anderen als die angegebenen Hilfsmittel - insbesondere keine im Quellenverzeichnis nicht benannten Internet-Quellen - benutzt habe und die Arbeit von mir vorher nicht in einem anderen Prüfungsverfahren eingereicht wurde. Die eingereichte schriftliche Fassung entspricht der auf dem elektronischen Speichermedium (CD-Rom).

	Ja	Nein
Mit der Einstellung der Arbeit in die Bibliothek bin ich einverstanden.	■	□
Der Veröffentlichung dieser Arbeit im Internet stimme ich zu.	■	□

Koblenz, 03.07.2019 Abdullah Imad Abdullah Elkindy

.....

(Ort, Datum) (Unterschrift)

Abstract

The status of Business Process Management (BPM) recommender systems is not quite clear as research states. The use of recommenders familiarized itself with the world during the rise of technological evolution in the past decade. Ever since then, several BPM recommender systems came about. However, not a lot of research is conducted in this field. It is not well known to what broad are the technologies used and how are they used. Moreover, this master's thesis aims at surveying the BPM recommender systems existing. Building on this, the recommendations come in different shapes. They can be position-based where an element is to be placed at an element's front, back or to auto-complete a missing link. On the other hand, Recommendations can be textual, to fill the labels of the elements.

Furthermore, the literature review for BPM recommender systems took place under the guides of a literature review framework. The framework suggests 5 stages of consecutive stages for this sake. The first stage is defining a scope for the research. Secondly, conceptualizing the topic by choosing key terms for literature research. After that in the third stage, comes the research stage. As for the fourth stage, it suggests choosing analysis features over which the literature is to be synthesized and compared. Finally, it recommends defining the research agenda to describe the reason for the literature review. By invoking the mentioned methodology, this master's thesis surveyed 18 BPM recommender systems. It was found as a result of the survey that there are not many different technologies for implementing the recommenders. It was also found that the majority of the recommenders suggest nodes that are yet to come in the model, which is called forward recommending. Also, one of the results of the survey indicated the scarce use of textual recommendations to BPM labels. Finally, 18 recommenders are considered less than excepted for a developing field therefore as a result, the survey found a shortage in the number of BPM recommender systems. The results indicate several shortages in several aspects in the field of BPM recommender systems. On this basis, this master's thesis recommends the future work on it the results.

Contents

List of Figures

List of Tables

1 Introduction and background

Business Process Management (BPM) is the discipline of handling organizational processes for the sole purpose of analyzing, designing, implementing and continuous improvement. In details, this definition evolved over the years ever since a spark back in the eighties lead to focus on the enhancement of the already existing business processes. Approaches like Total Quality Management (TQM) and processes improvement provided industries with more cost efficient, quality processes. Consequently, boosting the levels of revenue and customer satisfaction. A decade later, researchers started to shift focus onto Processes Re-engineering, given the fast growth of information technology and its potentials. Moreover, the research then took a deep look into the existing processes and searched for new ways to fundamentally redesign them to match the innovative changes in the field and the organizational goals. Nowadays, contemporary Business Process Management (BPM) sits on combining both approaches where in turns both processes effectiveness and efficiency are important vom Brocke & Rosemann (2014).

This development brought about further developments in other areas in our daily lives. The improvement of information technology (IT) brought about various functionalities to facilitate, utilize efforts by automating, simplify functionalities that in the old days required longer time and more effort. The solutions creators started to utilize this development resulting in applications like computer programs for companies, universities, schools and even personal use. Following the computer development wave, another wave hit a decade ago with the wide spread of smartphones and internet. Software technologies giants began to take an approach to their development that set an importance for the program's user usability of their programs that equals if not exceeds the actual functionality of the application. Therefore, the solutions became equipped with user specific behavior, where the application shows the user the information that matter to them. Another strong functionality that is newly introduced in our daily world is recommender systems. In details, the recommender systems suggest to the user information that fits their behavior/need. Moreover, examples to that can be found in the recommended videos to watch on YouTube, the search suggestions in Google or suggested friends to add on Facebook. On a similar note, recommender systems can be found in other less mainstream areas. An example to that is industry/ enterprise related areas in general and Business Process Management in particular. Further on, the topic of combining Business Process Management (BPM) with recommender systems is a fresh topic, yet a rich one. The following subsection introduces the use of recommender systems in Business Process Management (BPM). On one side, it elaborates the need for recommender systems. On the other side, it lists the ways a recommender system can be applied in Business Process Management BPM.vom Brocke & Rosemann (2014)

1.1 Business Process Management recommender systems

BPM is modeled with different languages. Most popularly are Business Process Model and Notation (BPMN) and Event-driven Process Chain (EPC) each of both suitable for certain business use. In details, EPC is convenient for high-level business processes description. Therefore, it captures model outcomes, risks, and issues. On the other hand, BPMN is more low level, consequently dealing with the processes themselves and engineering them to find a solution to business process problems. Although both notations differ in usage-purpose, however, they both share one major concept in common. Both notations are graphically represented relations of concepts. The figures 1.1 and 1.2 show a simple example of EPC and BPMN.

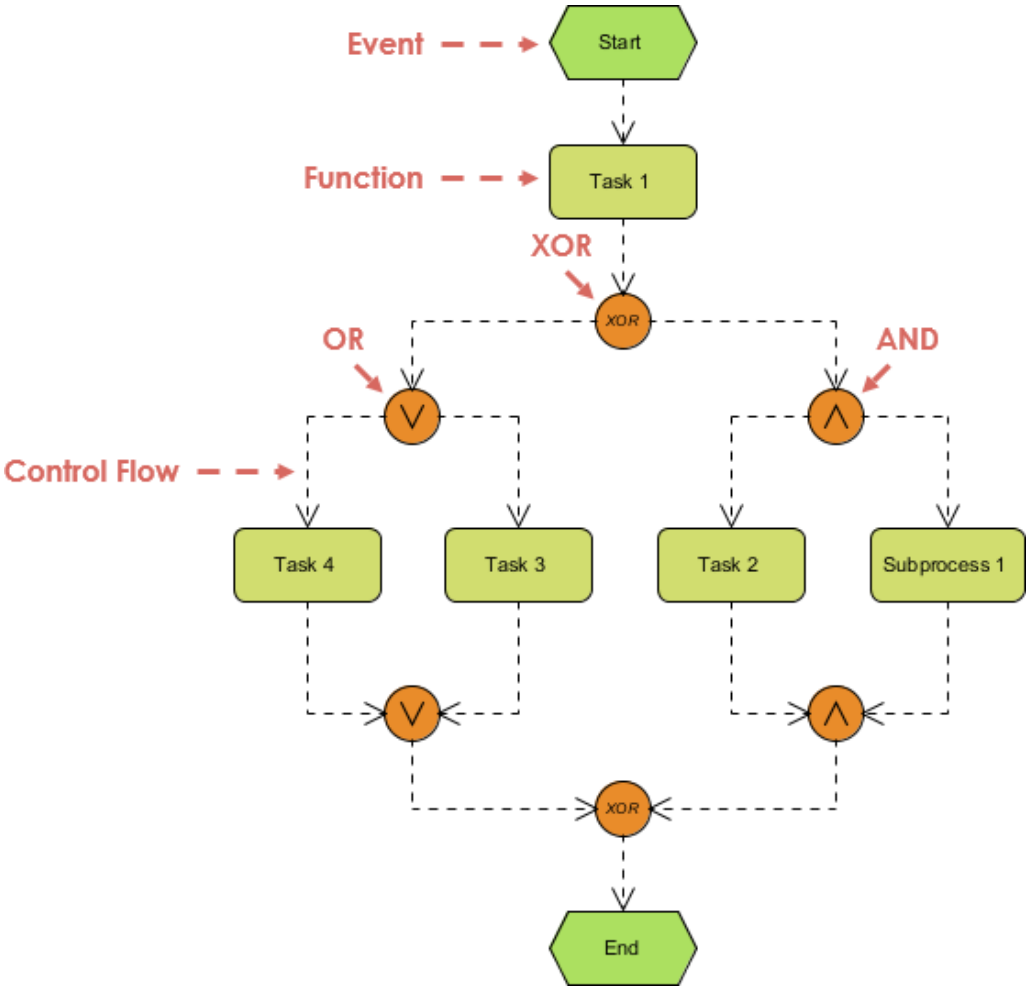


Figure 1.1: Example of EPC models
Source: https://circle.visual-paradigm.com/simple-template/

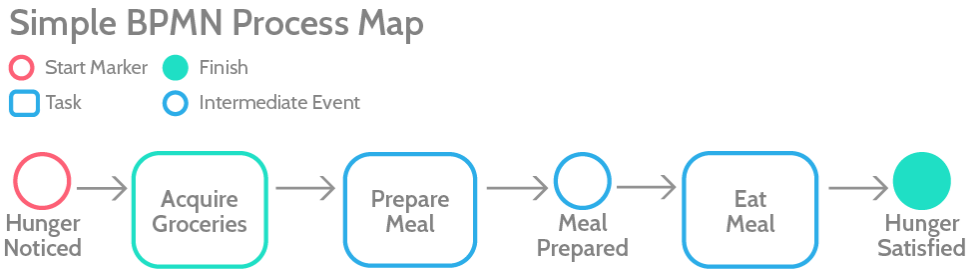


Figure 1.2: Example of BPMN models
Source: https://www.process.st/bpmn-tutorial/

1.2 Background

1.2.1 Why recommender systems are used in Business Process Management (BPM)

The nature of the visual presentation of Business Process (BP) models brought about several drawbacks that pushed the researchers to investigate support solutions to overcome them, support that came in the form of recommender systems. The following points describe how using recommender systems is beneficial in Business Process Management (BPM): Time and effort: modeling complex Business Process (BP) models can be time and effort consuming operation, especially for novice users who are not used to the syntax and recurring modeling problems. Using a recommender system reduces time and effort required for modeling. Error-prone model: combining graphical representations with complex business problems may introduce errors that the user's eyes can look over. Having a recommender system in most cases checks and validates the models under construction. Consequently, eliminating model bugs.

1. The Optimum model: a Business Process (BP) can be modeled in several ways, but not all of them represent the optimal ones. The model's quality depends on the users that create it. Experienced users develop more optimal, strong and adequate models. However, novice users' model can be less optimized and more error-prone. Moreover, recommender systems in most cases output the most fitting action for the process being modeled. Thus, using a recommender system helps novice users in the modeling process.

2. Time and effort: modeling complex Business Process (BP) models can be time and effort consuming operation, especially for novice users who are not used to the syntax and recurring modeling problems. Using a recommender system reduces time and effort required for modeling.

3. Error-prone model: combining graphical representations with complex business problems may introduce errors that the user's eyes can look over. Having a recommender system in most cases checks and validates the models under construction. Consequently, eliminating model bugs.

1.2.2 How recommender systems are used in Business Process Management (BPM)

So far, this paper discussed recommender systems and how they are beneficial for Business Process (BP) modeling. However, it's still not very clear how these recommendations look like. In details, recommendations for Business Process (BP) models are split into two main categories Textual and Overall recommendations. 1) Textual recommendations represent naming suggestions to elements like activities, events, gateways or even a whole model. The naming suggestion can be inserted in new elements or rewritten in already existing ones. These recommendations are not entirely popular but they exist. As for the 2) Overall recommendations, they cover a broader share of the existing recommenders. Overall recommendations are presented in relation to the structure of the recommenders like adding and subtracting elements from the Business Process Models.

Furthermore, the study in Kluza et al. (2013) suggests a classification schema to the recommendations where recommendations are classified in details into two types: Subject-based and Position-based recommendations. Subject-based recommendations are concerned with the element itself while Position-based recommendations focus on the place where the suggestion is needed. Moreover, both of them are divided into more types of recommendations. Figure 1.6 summarises the different recommendations types.

1. Subject-based classification: In Subject-based classification, the focus is driven towards the element. This leads the recommendation to be element based instead of where it is in the model. The recommendations like Attachment, Structural and Textual recommendations fall under this classification umbrella.

    a) Attachment recommendations: these recommendations suggest to users how to connect (attach) external entities to the element in hand. Moreover, these external entities can be in the shape of Decision Tables, Links, Service Tasks, and Sub-processes.

        i. Decision Tables: A Recommendation, in this case, is to connect the element, which is usually a gate to a decision table describing conditions.

        ii. Links: As for this case, recommendations suggest connecting to an intermediate Link Event.

        iii. Service Tasks: Service Tasks are recommended to the element in this case.

        iv. Sub-processes: In this case, sub-processes are recommended to be connected to the element in hand. This element is an activity.

    b) Structural recommendation: Structural recommendations tackle the element being suggested. Structurally speaking, a suggested element can either be Single Element or Structure of Elements.

i. Single Element: In this scenario, a single element is suggested to the user. In details, this element can be an activity, a gate, a swim-lane, an artifact, a data object or an event.

ii. Structure of Elements: As for this case, the user receives a whole connected structure of elements as suggested. A structure containing two or more connected elements.

c) Textual recommendations: Textual recommendations directly relate to a text-suggestion that the user receives for a certain element or ,in some cases, for the whole Business Process (BP) model.

i. Name of an Element: Includes textual recommendations for an element (activity, swim-lane or event).

A. Name completion: In name completion, the recommendation completes an existing element's label. For example, if an element's label is "Receive Data", a name completion textual recommendation can be as "Receive Data from Customer".

B. Full name suggestion: Full name suggestion on the other hand, recommends to the user a full element's label to overwrite.

ii. Guard Condition: Guard Condition suggestions are concerned with textual suggestions to gateways. Due to the importance of gateways in being a connection hub with different parts of the model, their labels need to make acquire proper meaning. Therefore, the guard condition suggestions need to go through a semantic analysis to better match the Business Process (BP) model.

2. Position-based classification: As for the second type of recommendations, Position-based classified recommendation deals with where the suggested element is to be positioned in the Business Process (BP) model.

a) Forward completion: Forward completion suggests to the user a new element to be placed in the proceeding part of where the suggestion is requested in the BP model. Figure 1.3 gives an example of Forward completion.

b) Backward completion: Opposite to Forward completion, Backward completion suggests to the users a new element in a succeeding position to where the recommendation is requested. An example for Backward Completion is figure 1.4.

c) Auto-completion: As for the last type of Position-based classified recommendation, Auto-completion suggests elements to auto-complete a Forward and a Backward BP parts. Figure 1.5 explains Auto-completion.

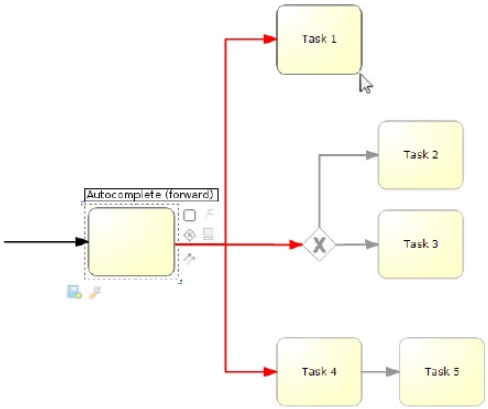


Figure 1.3: Forward completion
Source: Kluza et al., 2013

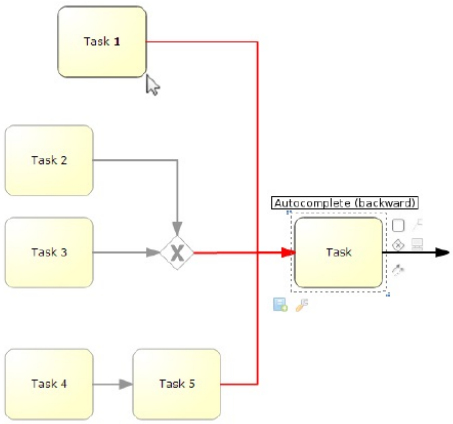


Figure 1.4: Backward completion
Source: Kluza et al., 2013

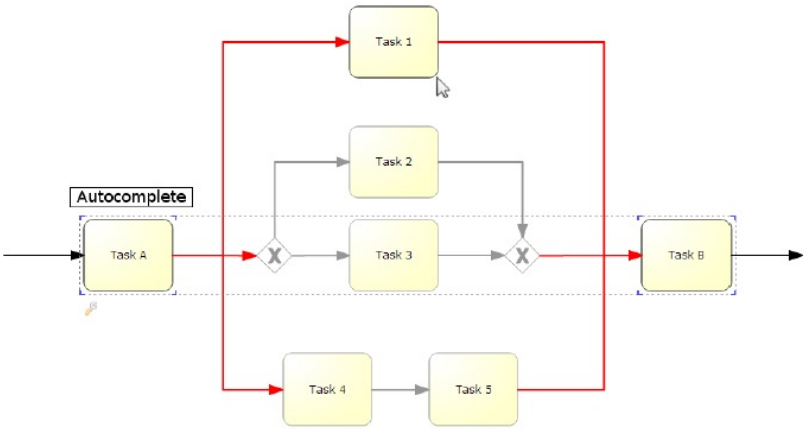


Figure 1.5: Auto-completion
Source: Kluza et al., 2013

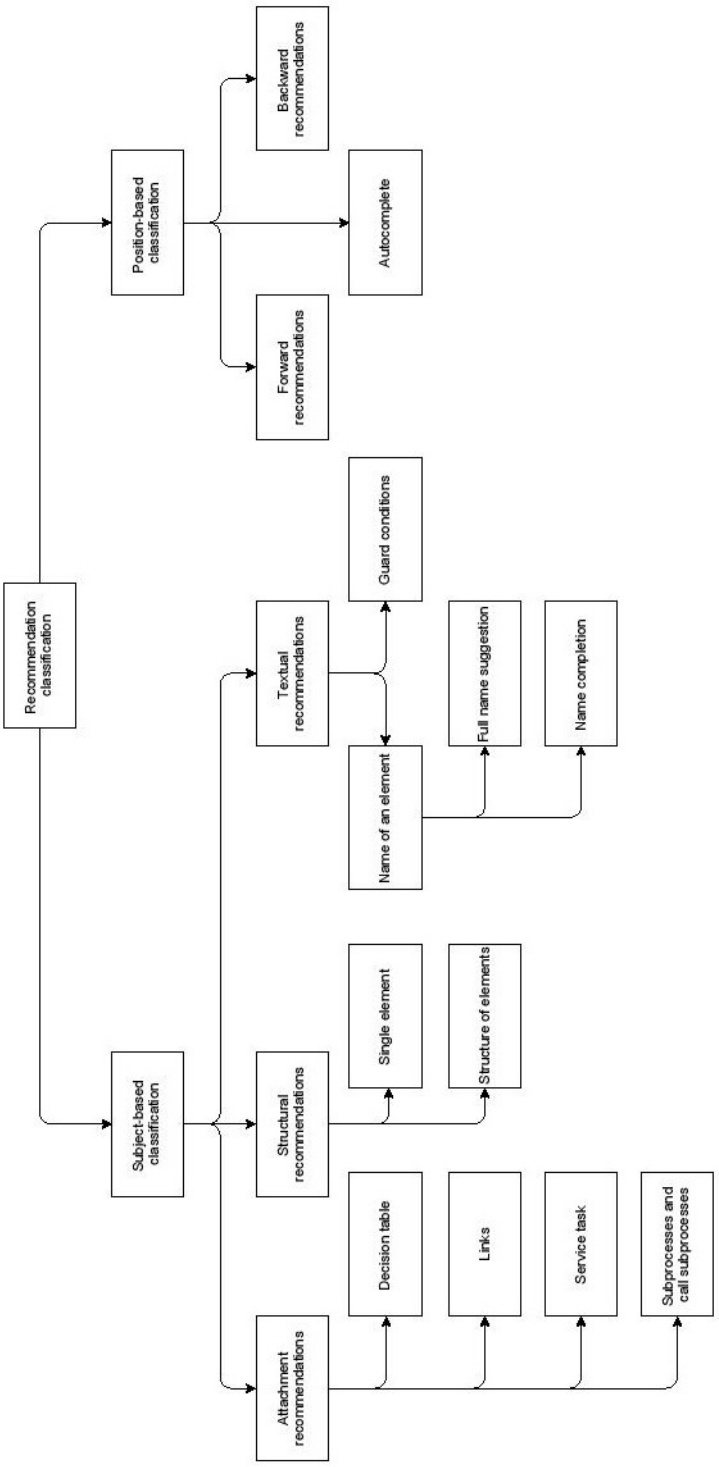


Figure 1.6: Types of recommendations
Source: Kluza et al., 2013

1.2.3 The need for a Business Process Management recommender systems survey

Building on what was previously explained, there are several tools and publications that have been developed in the market to support Business Process Management (BPM). This support, presented in BPM recommender systems, is implemented with different methodologies. Technologies like machine learning, graph mining and semantic context reasoning are examples of implementation methodologies to existing BPM recommenders. The recommendation process mainly consists of three main elements:

1. Reference Model: it is a BPM model in which the recommendation request takes place.

2. Models Database: In order to give a recommendation, a database containing examples of other Business Process (BP) models needs to exists or in some other cases, event logs of past processes usages are utilized. Moreover, these models cover different aspects of life like industry, health, insurance, etc.

3. Matching Mechanism: The Reference Model is matched to a corresponding model inside the Models Database. The Matching Mechanism differs between a model and the other. Some are with machine learning, others are with graph mining or semantic context reasoning. After the matching, the corresponding recommendation is given to the user, depending on the recommender itself.

The matching mechanisms are mainly what differs between BPM recommender systems in addition to the types of recommendations, explained in the previous sub-section.

Although there are rather vast differences between the BPM recommender systems, but it is not clear to what extent they are. Not all technologies in which the BPM recommenders are built with are listed to be compared. Thus, this is what this master's thesis sets to perform. It collects all the existing BPM recommender systems and compares them according to specified schematic methodology. This comparison derives a clear picture of all the BPM recommender systems out there which in terms is beneficial for a different reason. These reasons are as follows:

1. The types of different recommendations given for each recommender are known. This is informative for selecting a recommender system fitting a certain Business Process (BP) need.

2. Gathering up all the recommender systems and their implementation technologies creates clusters of recommenders where each group of similar recommenders resides together. This helps in detecting developments patterns, in addition to further comparisons on each cluster.

3. The comparison results show what technologies are in fact not used or lacking in the market.

4. This built up cumulative information opens the doors for creating more advanced BPM recommender systems either by combining existing technologies together or using technologies not used yet.

Finally, this thesis consists of 5 chapters. Chapter 1) introduces the topic of BPM recommender systems, the types of recommendations and the need for a BPM recommender systems survey. As for chapter 2), it elaborates the methodology implemented. In details, it describes the BPM recommender systems research process, in addition to the analysis comparison schema which the different BPM recommenders are compared with. Chapter 3) delves deep into the comparison between each recommender by giving a summary of the recommenders and comparing them with each other in analysis comparison tables. Moreover, chapter 4) discusses the BPM recommender systems comparison results. It suggests further development approaches as well. Finally, chapter 5) concludes the overall presented information and summarizes it.

2 Methodology

The topic of BPM recommender systems is rather specific. It is concentrated in certain fields and themes. Abstractly speaking, investigating this field requires good quality knowledge about the existing technologies. And this, in turn, indicates a need for a literature review process with clear structured guided points to maintain strong informative literature review quality. The literature review quality sets the effectiveness of the research process, and consequently, the status of the BPM recommender system publications reviewed.

Furthermore, this chapter presents the research process that took place for this thesis. It explains thoroughly how the BPM recommender systems publications were researched, filtered, reviewed and compared to one another. In addition, this chapter presents the features in which the BPM recommender systems are to be compared with.

2.1 Literature review framework

The information about BPM ,in general, and BPM recommender systems ,in particular, all lay under the topic of Information Systems (IS). Information Systems (IS) exist in different publications coming from different sources like articles, databases and, journals. However, they are focused in certain aspects. Therefore it was suggested to use a themed literature review framework to handle this type of information. The framework groups the scattered information together and gives a precise and more topic-specific utilized result.

The authors vom Brocke et al vom Brocke et al. (2009) propose a framework for conducting literature review. The proposed framework focuses on the searching process of literature, thus establishing good grounds for solid information quality. Moreover, it consists of 5 sequential stages as shown in the following figure. The stages are: 1) definition of review scope, 2) conceptualization of topic, 3) literature research, 4) literature analysis and synthesis and finally 5) research agenda. This framework was implemented to conduct the literature review for this thesis. The following sub-section delves deep into the stages of the framework.

As mentioned earlier, the framework provides structural instructions to produce high quality literature through. It carries out this task through a sequence of 5 stages. For the sake of this master's thesis, most of the stages were implemented as is. However, few were slightly altered and extended to better fit the theme and topic of BPM recommender systems and its literature. Figure 2.1 presents an overview of the framework's stages. Moreover, this section explains each stage as proposed by vom Brocke et al vom Brocke et al. (2009). Simultaneously providing the extension selected for this thesis and how each stage is implemented for BPM recommender systems.

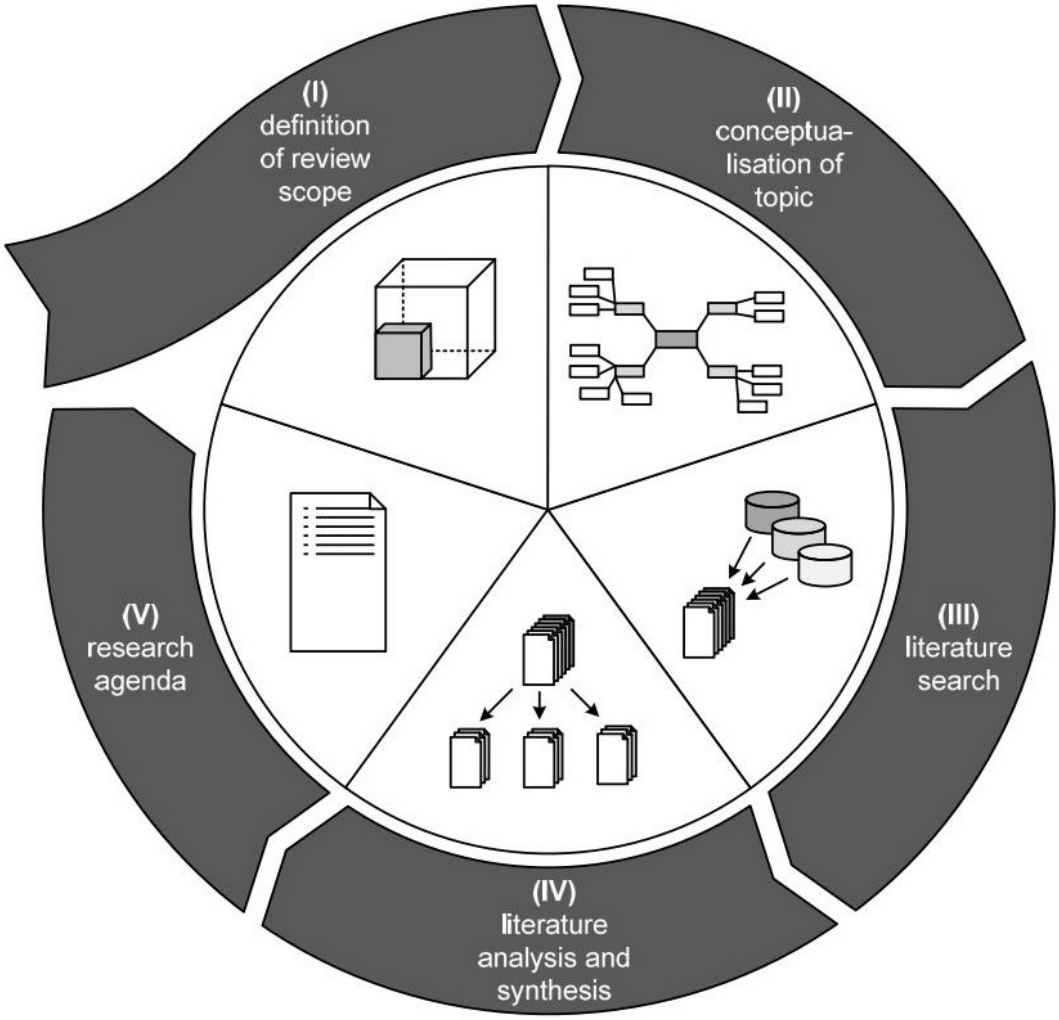


Figure 2.1: Framework for literature review
Source: vom Brocke et al. 2009

2.1.1 Definition of scope

The scope confines the literature research to defined borders. Which in turn, paints a clear picture of the literature. Furthermore, the scope defines why the research is done, what exactly the research conducted is in addition to other information. The framework proposes 6 characteristics to be set to define the scope of the literature research. The characteristics are as follows: 1) research outcomes, 2) goal, 3) organization, 4) perspective, 5) audience and 6) coverage. Furthermore, the values to the characteristics are classified into categories, each characteristic containing a group of categories to fit its values. Table 2.1 shows the characteristics-categories relationship. It elaborates each characteristic and its corresponding categories. The categories with different background represent the value implemented in this thesis. Moreover, the characteristics are in details as follows:

1. Focus- is where the research concentrates. It has a value of 4 categories:

    a) Research outcomes- focuses on the result of a research

b) Research methods- searches for the way a certain publication is implemented.

c) Theories- investigates general theories.

d) Applications- searches for the applications, in general, explained in the research publications.

The research for BPM recommender systems and comparing them requires a focus on the "Applications" for each system. Therefore, it was chosen for this purpose.

2. Goal- elaborates the aim of the research. The goal of the literature is between the following 3 categories:

   a) Integration- represents the aim of the literature research where different literature is to be group with one another and integrated.

   b) Criticize- is the aim to conduct literature research to criticize the existing publications.

   c) Summarizing- the aim of some literature research is to summarise the publications.

   Moreover, comparing BPM recommender systems to one another confluences with the aim of "Summarizing" literature to be compared.

3. Organization- indicates the way of the literature review is viewed. Organization consists of 3 categories:

   a) Historical- implies the organization of literature review according to the publications timeline.

   b) Conceptual- on the other hand, relates to organizing the literature review against the concepts being discussed.

   c) Methodological- is concerned with organizing research according to the methods each publication presents.

   Furthermore, "Methodological" organization was chosen to define BPM recommender systems literature research. The methods a BPM recommender system is implemented is explained directly using methodologically organized literature review.

4. Perspective- is the representation of the literature review. It indicates the author's stand from the publications. Moreover, the perspective in the scope definition is divided into two categories:

   a) Neutral representation- where the author represent the information in the literature review in a neutral stand. Here the author merely represents the data as they are without a stating a position concerning it.

   b) Espousal of position- on the other hand, represents having a position concerning a matter and reflecting it on the literature review.

In comparing BPM recommender systems literature review, there is no need to take a position concerning the matter. On the contrary, reviewing the difference requires a neutral stand. There "Neutral representation" was selected for this thesis's literature review.

5. Audience- sets the type of receivers that the literature review is directed to. This exclusively impacts the writing style of the author. Nevertheless, there are 4 categories for the audience:

   a) Specialized scholars- are specialized audience directly concerned in the literature review with the matter. Much often, the topics presented to this audience, in this case, are specific and close to their field of knowledge.

   b) General Scholars- contractually are concerned with broader more general matters.

   c) Practitioners /Politicians- indicate the audience of politicians or practitioners who require practical publications in their field.

   d) General public- are the average people who range from young students to old adults.

   The topic of BPM in general and BPM recommender systems, in particular, is very specific. Therefore, "Specialized scholars" were selected as the target audience of this master's thesis,

6. Coverage- represents the degree of sources coverage in the literature review. In other words, it indicates to what extent are the publications in a certain matter are covered. The coverage characteristic consists of 4 main categories:

   a) Exhaustive- research includes the entire literature in a topic or the majority in it.

   b) Exhaustive and selective- research considers exhaustively all the literature on the topic but represents a selected portion of it.

   c) Representative- research on the other hand, namely includes samples to considers and review. The sample, in this case, is considered a representation of the whole literature.

   d) Central/ Pivotal- research considers reviewing only the pivotal literature for the topic.

   "Central/ Pivotal" coverage was considered for BPM recommender systems due to how specific the topic is. Both topics BPM and recommender systems are considered rather general topics in the field. However, combining them results in a very specific topic.

Characteristic		Categories			
1	Focus	Research outcomes	Research methods	Theories	Applications
2	Goal	Integration	Criticism	Summarising	
3	Organization	Historical	Conceptual	Methodological	
4	Perspective	Neutral representation		Espousal of position	
5	Audience	Specialized scholars	General scholars	Practitioners/ Politicians	General public
6	Coverage	Exhaustive	Exhaustive and selective	Representative	Central/ Pivotal

Table 2.1: Scope definition table

To summarize the scope defined for this thesis, the literature review focuses on BPM recommender system "Applications" with the goal of "Summarizing" the gathered literature. Moreover, it organizes the literature "Methodologically", describing it in a "Neutral representation" for "Specialized scholars". Finally, only "Central/ Pivotal" publications are selected for the review.

2.1.2 Conceptualization of topic

In the next phase, a broad conception of the topic and information concerning it is needed. The broad conception of the topic results in a good understanding of the topic including its fields of application, methodologies, and key terms. Moreover, the key terms in addition to the other inquired information are used in the coming phase, the research phase. The proposed framework suggests exploring publications which contain a summary or an overview of the topic. The broad conception these publications present provides the literature research strong, topic related key terms. In addition, the frameworks recommend broadening the key terms with the use of relevant key terms, synonyms, and homonyms. During the conceptualizing of BPM recommender systems, the key terms had to be precise and effective due to the "Central/ Pivotal" coverage of the literature review. Therefore, a paper presents an overview of BPM recommender systems was read through [2] and key terms were extracted. The term "BPM recommender systems" was a strong key term that was extracted. Furthermore, with the use of linguistic and topic related terms, an additional key term was extracted "Recommendation based bpm". The key terms updated if new ones are found. For BPM recommenders, further key terms were discovered in the coming phase during research. This is explained in the next point.

2.1.3 Literature research

Literature research is the pivotal phase during literature review. It collects the information to be discussed. In general, strong literature research ensures good quality literature. Abstractly, the research process begins with key terms research, extracted in the latter phase. The result of the search is presented to forward and backward search. In details, the process of backward search is concerned with the literature cited within a certain publication. More often known as the publication's references. On the other hand, a forward search for publication "A" is concerned with reviewing the publications that cite it. Moreover, these two functionalities result in a large pile of literature. Some of the literature is very relevant, while others are not. Some are weaker in quality, in comparison to the others. Therefore, ongoing quality evaluation is required. The evaluation limits the literature collected, from key terms, forward and backward search, to only the publications that are directly related to the topic.

For BPM recommender systems, the research process started with gathering up literature by the key terms "BPM recommender systems" and "Recommendation related BPM" search. The result was 8 publications ranging in quality and relevance to the topic. After that, a forward and backward search was executed on the 8 publications resulting in many publications. However, going through the found publication titles introduced new topic related terms and concepts. Therefore, the key terms were updated introducing key terms like "Semantic-based BPM", "BPM model recommendation", "Textual recommendations in BPM", "Machine learning based recommender bpm" and "BPMN auto-completion". The final result compromised of the latter forward and backward research and the key terms search of the newly introduced key terms. The publications mounted to 240 publication. Reflecting on the scope of this thesis. The "Central/ Pivotal" coverage requires topic specific literature. Therefore, the 240 publications required a thorough evaluation to ensure that the remaining ones of them are directly concerned with BPM recommender systems. Furthermore, the evaluation process implemented consisted of 3 main sequential tasks:

1. Title analysis: it implies analysing the publication's title for BPM recommender system related keywords. The titles holding BPM in addition to recommendations and their methodology are most fit in the valuation.

2. Keywords search: during this task, the publication is searched for related keywords. Moreover, the keywords are not the key terms that were extracted in the latter section but they are directly concerned with BPM recommender systems. Example to them is "BPM", "Business Process Management", "Recommendations" and "Suggestions".

3. Abstract and implementation review: after analysing the title and looking for keywords, the abstract of the publication is read and analysed. If it contains an indication that it is concerned with BPM recommendations or assistance, then the implementation section is reviewed to confirm if the publication presents a BPM recommender system.

Filtering the publications through these tasks resulted in 18 publications that were chosen to be the main literature for BPM recommender systems.

2.1.4 Literature analysis and synthesis

After looking thoroughly through the literature databases, with the literature review scope and key terms in perspective, the collected literature moves to the main phase "Literature analysis and synthesis" during which the collected publications are reviewed in details. Moreover, this literature review framework suggests setting units of analysis on which each publication is to be reviewed upon. In other words, the publications will be synthesised and compared to one another according to comparison analysis features. After setting the features, it suggests building a comparison matrix where all the literature publications are listed and their corresponding feature analysis are checked. Table 2.2 describes the comparison tables called the concept matrix. On the left, reside the publications while on the right, the analysis features stand. The features close to each other are grouped by concepts. If a publication fits a certain feature, then the corresponding field is marked.

Publication	Concept						
	A			B			...
Analysis feature	C	D	E	F	G	H	...
Publication 1	X			X			...
Publication 2		X		X			...
Publication 3			X			X	...
...							...

Table 2.2: Concepts matrix example

For this literature review, the features were set according to the common knowledge about technologies implemented in recommender systems and BPM. In addition, during the phase of "Literature research", skimming through the found literature introduced other common features which were not intuitive to include from start. Therefore, all the gathered features were grouped and set to be used in the comparison analysis. The following subchapter "Comparison analysis features" lists the features selected in details.

2.1.5 Research agenda

Lastly, the research agenda distinguishes what the author is pursuing to achieve with the literature review. It achieves that by comparing the recommenders to one another on the basis of features. For example, a feature like the method that the recommender is implemented with, or the type of recommendation it provides are a strong output of the analysis comparison. Moreover, This analysis brings about an informative overview of the technologies in BPM recommenders. Moreover, the conception matrix provided analysis

features where little work, or at few no work, is done. Thus, exposing the methodologies where BPM recommender systems have a shortage. Therefore, the agenda in this master's thesis aimed at providing the field of BPM recommender systems' future work a broad overview of the technologies existing, what they conform of and where they have shortages.

2.2 Comparison analysis features

The last section introduced thoroughly the framework that this thesis has implemented for performing the literature review for BPM recommender systems. It described the review as a descriptive comparison process where BPM recommender system publications are compared to each other according to certain features. This section delves deep into these features. It lists them, groups them, and introduces a description explaining them.

In this master's thesis, the following comparison analysis features were set according to knowledge about recommender systems acquired through common knowledge and during the phase of "Literature research" that was performed. Skimming through the BPM recommender systems publications introduced a wider list of features. Moreover, the comparison analysis features are:

1. Method- indicates the way the recommender is implemented. There are several ways recommenders in general are made with. Moreover, the following are the values the features "Method" order alphabetically:

    a) Association rule mining- method is based on the approach which utilizes association rule mining. Association rule mining is a rule-based machine learning algorithm which associates closely related variables together. Moreover, the closely related variables can be used, in BPM recommender systems, to express the BP process parts that are close to each other, therefore supporting generating a fitting recommendation.

    b) Based on history- method refers to recommenders that are implemented with approaches based on the modeling history. In other words, it is set for a recommender which utilizes the modeling environment's usage history to suggest new recommendations.

    c) Bayesian networks- method is mainly concerned with the probabilistic graphical model Bayesian networks. A Bayesian network provides probabilities for its nodes. Moreover, the close patterns to be recommended can be generated using the aforementioned network probabilities.

    d) Deep learning/ neural networks- methods are concerned with deep learning and neural networks. Many recommenders, not only in BPM, utilise these methods due to their ability to learn and find paths and patterns.

e) Graph mining- method indicates the approach which mines BP models for graph patterns to be used as further recommendations.

f) Notation independent model- method indicates an approach where BPM recommendations are brought about using an original model that is BPM notation independent. In other words, the approach can be applied to any BPM notation without taking its notation into account.

g) Semantic method- supports BPM modeling by recommending suggestions to users using a semantic model. The semantic models enrich the BP models with context awareness, thus providing them with sense. The sense that in turns can be used to recommend to the users fitting suggestions to their models.

h) Support Vector Machine- method is based on transforming BP models into vectors to be compared for similarity in the SVM space like Tf-Idf for example. The closer models can be used for recommendation.

i) Syntax-based assistance- method uses syntax proposed solutions to generate recommendations. The BP models syntax can be taken into a model that describes them syntactically, consequently providing options to give recommendations.

j) Textual model- method indicates a recommender system that uses an original approach to recommend textual recommendations in BP models. Textual recommendations are concerned with the textual content of models' labels.

2. Notation: The main graphical modeling notations for business process management are BPMN and EPC as explained in the first chapter. In addition to that, other modeling notations are used in BPM, not as frequent though. Therefore, the following notation features were chosen to set if a recommender's publication has mentioned them:

   a) BPEL

   b) BPMN

   c) EPC

   d) Petri-nets

   e) XML

3. Recommendation type: The first chapter, introduced the different types of recommendations which users can receive. For the sake of the comparison analysis, notation features appearing in the publication are checked. Moreover, a BPM recommender system publication can contain multiple features at the same time. The following are the features group by recommendation type:

   a) Attachment- recommendations conform to the following analysis features:

i. Decision table

ii. Link

iii. Service task

iv. Sub-processes

b) Structural- recommendations, on the other hand, has two features:

i. Single element

ii. Structure of elements

c) Textual- recommendations has contains types of features:

i. Fullname suggestion

ii. Name completion

iii. Guard condition

d) Position-based

4. Form: It indicates the form of the recommender mentioned in the publication. Moreover, 3 features are proposed for this sake:

a) Application- feature means that the publication proposes a BPM recommender system application.

b) Extension/ plugin- feature on the other hand, is set if the publication introduces the recommender as an extension or plugin to an already existing BP modeling environment.

c) Framework/ concept- feature is concerned with publications that introduce a concept or framework for providing BPM recommendations.

5. Repository: Each recommender works by checking a BP models repository, to assist it in suggesting recommendations to users. The models residing in these repositories provide valuable patterns that can be further used in the recommendation process. Moreover, there are two types of repositories:

a) Event logs- are used by recommenders that log their BP models path calls. These logged event paths are later used as a repository for recommendations.

b) Models database- are on the other hand an existing database of models.

6. Common properties: The common properties are common properties noticed in the BPM recommender system publications. They are as follows:

a) Social features- feature is set if the recommender has social features that affect the recommendation like other users behavior for example.

b) Personalization- feature indicates that the recommender is aware of the user's behavior. Thus recommending him/her suggestions fitting their personal behavior.

c) Not domain specific- is set if the publication explicitly mentions that the recommender is not domain specific.

d) Accuracy/ confidence- indicates if the publication has accuracy and confidence check.

e) Recall/ precision- also indicates if the publication contains recall and precision check.

f) Evaluation check- is set if the recommender's publication uses an evaluation schema to validate the approach it is proposing.

g) Has an example- feature is concerned with BP recommender publications that contain an example where the recommender is applied.

h) Uses ontologies- indicates if the publication uses semantics in the process of recommendations generation.

i) Not fully automated- is set if the recommender's publication mentions that some parts are done by hand.

7. Utilities: It indicates what technologies, libraries or programming languages the recommender uses.

8. Distance measure: The majority of the BPM recommender systems utilise a distance measure metric to detect similarities between the model in need of suggestion (Reference model) and the repository.

3 BPM recommender systems survey

This chapter is the core of this master's thesis. It consists of two sub-chapters. For the first sub-chapter, it introduces the recommender systems in a detailed overview. The second sub-chapter on the other hand, presents the comparison analysis. It achieves that through a sequence of comparison tables. Moreover, the tables use the comparison analysis features mentioned in the last chapter.

3.1 BPM recommender systems review

The literature review found 18 recommender systems relevant to survey the BPM recommender systems. Moreover, these recommenders are listed in this sub-chapter with a detailed overview of each one. Furthermore, the overview describes the recommenders' implementation methods. In addition, it mentions the type of recommendations presented, the BPM notation used and other information.

3.1.1 A Recommendation System to Facilitate Business Process Modeling

This recommender Shuiguang Deng (2017) proposes a recommendation system that takes business processes and suggests to them which activity is to come next in the flow. It supports a different range of business process representations like Petri-nets, XML and business process execution language. The recommendation is brought about through two main phases: Offline Mining and Online Recommendation. Offline Mining takes place in the backend before starting to model. During Offline Mining, the recommender takes in processes inside repositories called Process Filest, and finds relations amongst their processes. This is carried out using graph mining. As for the second phase Online Recommendation, the suggested activities get propagated to the user by calculating the difference between the model under development (Reference Model) and the patterns extracted from the Process Files.

The creators of this recommender used a unified concept throughout the two phases of a process recommendation. The concept denotes that each BPM consists of a set of candidate nodes and an upstream subpath. The candidate nodes are the nodes on the last level of the model, which are usually the candidates to be recommender hence the name. As for the upstream subpath notation. It denotes the remain upstream path before the candidate notes. For example, if digraph(A->B; B->C; C->D;C->E) is a BP model then the set D, E contain the candidate nodes while the upstream subpath is digraph (A->B->C).

The latter concept is utilized throughout Offline Mining and Online Recommendation. During Offline Mining, the upstream subpath of each business process residing the Process Files is inspected for all possible subgraphs given the candidate set for that business process. Once these subgraphs are found, their corresponding confidence values are calculated over the whole Process Files. Thus, the user knows whats the confidence of a certain pattern to appear. Once the user starts to request recommendations, distance measurement calculations are applied between the mined subgraphs in Offline Mining and the upstream subpath for the model under development. The recommendation sets are produced to the user according to the distance and confidence of each subgraph.

3.1.2 Assisting Business Process Design by Activity Neighborhood Context Matching

This recommender Chan et al. (2012) suggests to users a list of relevant activities. The way it works is by choosing which nodes require recommendation for. After that, the recommender is invoked for suggestions. The system will output to the user a list of relevant activities for each node the user picked for recommendation. Technically, this system is built upon context aware systems, where a certain activity will have a context in which similar activities can be compared to. These recommendations can be utilized by adding them in the same zone they are required for recommending

The concept of a zone is exploited in this recommender as the main functionality that the authors used to build this system. A zone in a BP model is the position in which a certain activity resides. Zones are helpful to perform activities neighbourhood context matching using graph theory. The neighbourhood context matching calculations compare the zones between the node to be recommended to other nodes in other business processes. These business processes exist in a shared repository of BP models. The calculations are carried out through an original approach where the zones connections are taken into consideration. Each connection has a label. Therefore, it can be represented by a string of words. Moreover, the Levenshtein distance measure is used to check the similarity between zones connections. The authors propose to match all connections belonging to the same zone and that have the similar ending activities. The similarity check also covers more than one zone according to a parameter the user sets. For example, if the user requires a similarity check for an activity A throughout 3 zones. This means that the system will not only check the zone where the activity A resides but in the two succeeding zones as well. Consequently, the system implements a similarity check for the 3 zones throughout the whole repository for matching 3 zones that are similar. Furthermore, it is notable that the higher the number of zones coverage is, the smaller the number of output of course. According to an experiment the authors carried out on the system, whenever the number of zones to be checked passed 3, the collected tasks to be recommended were close to zero at cases when the similarity was high. This makes sense as there are not many graphs structures that will be very similar to each other if they are large. The following figure 3.1 shows the results of the evaluation carried out.

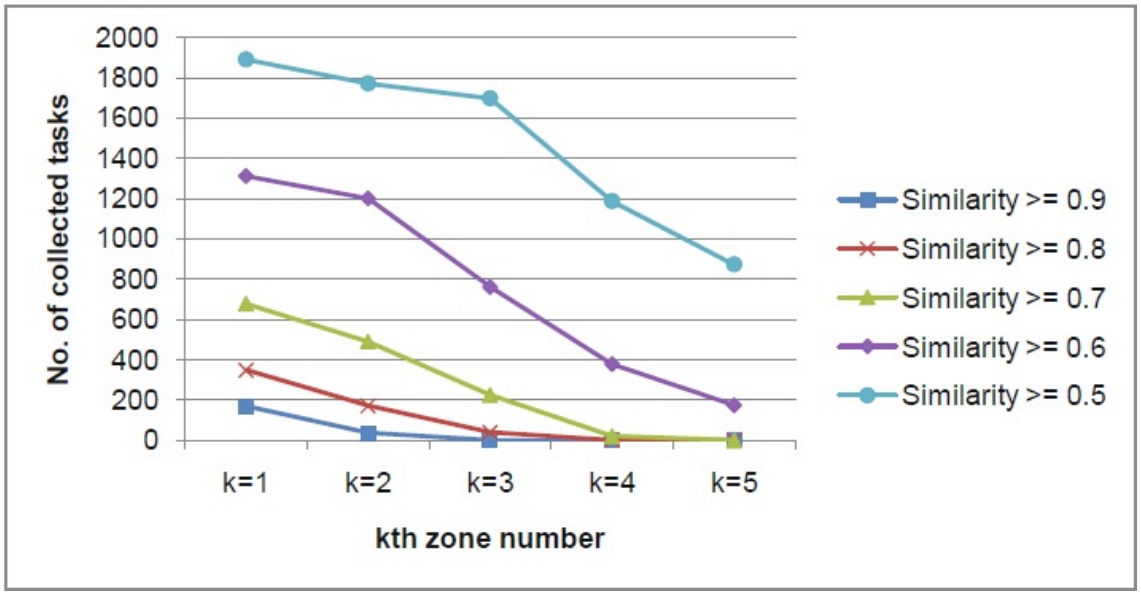


Figure 3.1: The evaluation's results
Source: Chan et al. 2012

3.1.3 Context-Based Service Recommendation for Assisting Business Process Design

The recommender proposed in this publication Nguyen et al. (2011) is a graph based context oriented recommender. In other words, the recommender uses the graphical context of the reference model and compares it to models contained in a repository. The user chooses the activity he wants the recommendation for. After the context based comparison, the user receives the closest activities to the one chosen activity. To model the graph context of a business process model, the authors proposed a modeling mechanism that will be explained in the following paragraph. Furthermore, this paper refers to the model activity as web service. As for the process model, it's defined as the service composition. There are two main coinciding factors that define a service composition "model" context:

1. Firstly, the direct links that point to a web service "activity". The direct link from a web service to the other inside a service composition declare a direct strong relation, thus utilizing it with defining the context of a service composition is beneficial. That is due to the fact that for example, a direct relation A->B in a service composition, if found in another service composition would relate both service compositions to one another.

2. Secondly, the layers of path depth distance from the selected web service "activity" and the remaining web services in the service composition "model". The following two figures show two service compositions in figure 3.2 and their inferred Composition context graphs in figure 3.3. In more details, the scenario in the following two figures is comparing the activities s2 and s6. The zones created created from the path distance of the selected web services is exploited in defining the context size. In other words, in order to define the context of a web service "activity", how many zones far away from it should be considered.

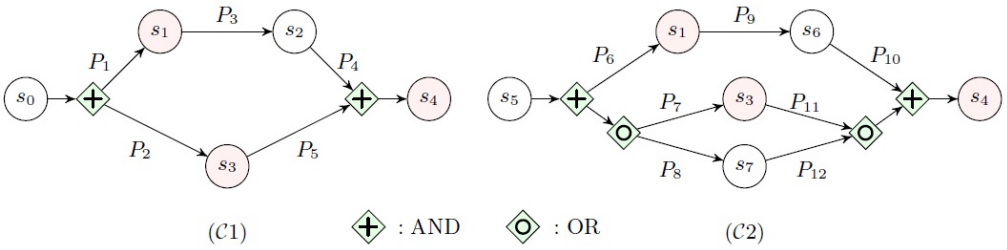


Figure 3.2: Service compositions example
Source: Nguyen et al. (2011)

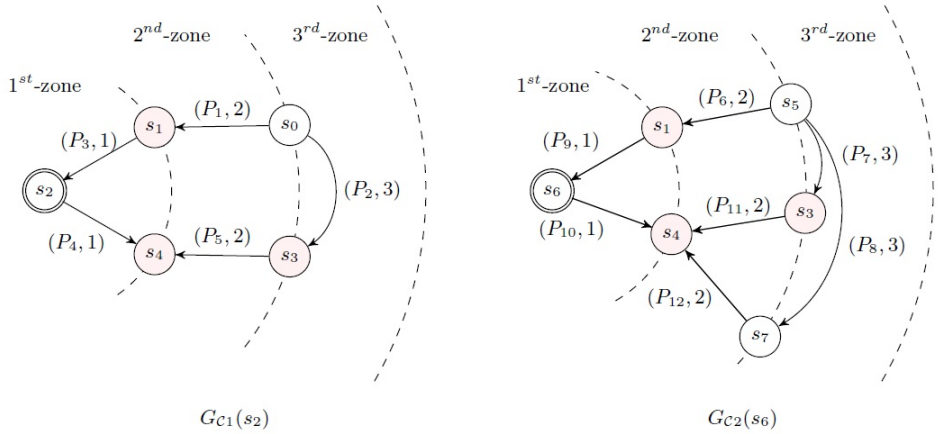


Figure 3.3: Inferred context graph
Source: Nguyen et al. (2011)

With that proposed modeling mechanism of defining the service composition context, the authors needed to propose a similarity check functionality where contexts can be compared to one another. As previously stated, the two main factors that this mechanism depends on rely mainly on the direct link between services, and the path depth of the selected service to the others inside the service composition. For handling the direct links, the authors performed a Levenshtein string distance measure. In more details, every direct link has a label describing it. For instance, looking at the figure above, the direct link P3 has a label value of "Sequence". Also, the label for the directly link P4 is "AND-join". Therefore, the authors proposed the string based distance measurement in order to capture the graphical similarity between the different service compositions. On the other hand, the path depth for the selected web service with the remaining services is used as mentioned before to set the size of the context. The similarity equation contains both the depth and the direct link Levenshtein distance. Lastly, the equation also contains weights that give more importance to the web services that are closer to the selected web service. Consequently, returning to the given example in the figure, the nodes s1 and s4 have higher importance than the node s3 for example, thus their weights will be higher.

To validate the approach proposed, the authors built a Java Applet to handle the recommendation processes. Data were gathered to form a services composition repository

and the business process notation supported is BPEL. According to the results, the authors found out that the proposed solution successfully supported users with similar web services. They also discovered that the higher the depth length is the lesser number of matched results. This effect is rather logical, due to the fact that the larger the context is, the less the chance it will reoccur in other service compositions.

3.1.4 Action Patterns in Business Process Models

This recommender system Smirnov et al. (2009) suggests to the user while modeling the next activity in the workflow stream. The core of this recommender revolves around a concept classified as Action Patterns. Action Patterns, contrary to well-established workflow patterns, deal more with the semantic context of Business Processes' content. In this notation, the Action refers to the syntactic verb existing in the BP activities' labels. For example, a BP containing an activity with the following label "Check the bill", will correspond to an action "Check".

Action Patterns change from a context to the other and from a domain to the other. But once, they are mined, they represent the recommendations for the current model at hand. Mining Action Patterns in this recommender utilizes association rule mining, thus retrieving the closely related actions together for the in-development BPM. The approach in this recommender supports both domain-aware and standalone BP models.

Action patterns mining goes through two consecutive stages: Co-occurrence Action Patterns mining and Behavioral Action Patterns mining. Co-occurrence Action Patterns represent actions that occur together in a process model collection. Process collections are mined using keywords in the actions labels. The result of this association rule mining algorithm is the co-occurrence action patterns. A drawback of this stage is that during association rule mining in finding co-occurrence action patterns, the order of the BPM activities, thus their semantics are not taken into consideration. That's where behavioral action patterns mining takes place. In this phase, the recommender defines relations between activities mined during co-occurrence action patterns mining. These relations are later passed on to an association rule mining algorithm, the same as the algorithm used in the previous stage.

The association rule mining algorithm implemented, takes into consideration the confidence and support of the resulting set according to a minimum support (minsup) and minimum confidence values (minconf).

The paper presenting this recommender evaluated it based on the SAP Reference Model, which is a large process model collection covering 29 divisions of an enterprise like sales, accounting and so on, containing around 604 Event-driven Process Chains (EPCs). The results retrieved were not as high in comparison to the number of processors in the process collection with an only maximum minsup value of 9. For example, during co-occurrence Action Patterns mining, the resulting set of a minsup of 7 with minconf of 95% is only 5 sets. It means, if it is wanted to be 95% confident about the possibility of a certain

activity combination, then out of all activities in the collection, only 5 combinations will be the result. This is due to the loss of semantics occurring from the usage of WordNet clustering. But the main reason for the this is the heterogeneous clusters inside the collection. Meaning that these sets will not occur in all the different domain process in the world but only in specific ones.

3.1.5 An Efficient Recommendation Method for Improving Business Process Modeling

A Petri-net business process model consists of nodes connected by edges serving a certain business goal. A node can either be a place "p" performing a certain task or a transition "t" combining two edges together. This recommender Li et al. (2014) suggests to users what the coming node is in their reference model. In this context, the authors hypothesized a framework where they consider that a BP model consists of a candidate node set and an upstream subgraph. The candidate node set consists of the nodes existing in the last level of the model. While the upstream subgraph represent the subgraph preceding the candidate node set. This recommender gives two types of suggestions, Forward and backward extension. They directly depend on the form of the candidate set. Forward extension, where a new place node "p" is recommended and connected to the candidate node, occurs when the candidate nodes set consists of only one node. While backward extension, it connects the candidate node with a node existing on the upstream subgraph.

Moreover, the recommender contains a shared repository of BP models that the users receive suggestion from. The main idea of this recommender is based on finding the closest model in the repository to the reference model under development. This is brought about through two phases: an offline and an online phase. During the offline phase, patterns are extracted using pattern mining techniques. Once the user asks for a recommendation, the upstream subgraph in the reference model is compared to the mined patterns' upstream subgraphs and a distance measurement is performed. The closest model in distance will in term be used for recommendation by suggesting its candidate nodes set.

Patterns are mined through an algorithm called gSpan which is an association rule mining algorithm. The nodes are traversed in a depth first search (DFS) manner. But, the patterns that could be extracted from one model could vary in form as shown in the figure 3.4. Moreover, this indicates that their corresponding DFS will be variant as well. Consequently, the DFS that will give the minimum cost is extracted. The cost that is calculated according to the lexographics of the nodes and edges.

Each pattern will be assigned to one DFS pattern with a min DFS trees and it will be represented in a string. To measure the distance of the min DFS trees and the reference model under development, three factors have to be taken into consideration:

1. minDFS code: each pattern's min DFS tree can be represented as a string. The following is an example of this string: +12-23. This represents for example and edge going from node 1 to node 2 and then a reverse edge going from node 3 to node 2.

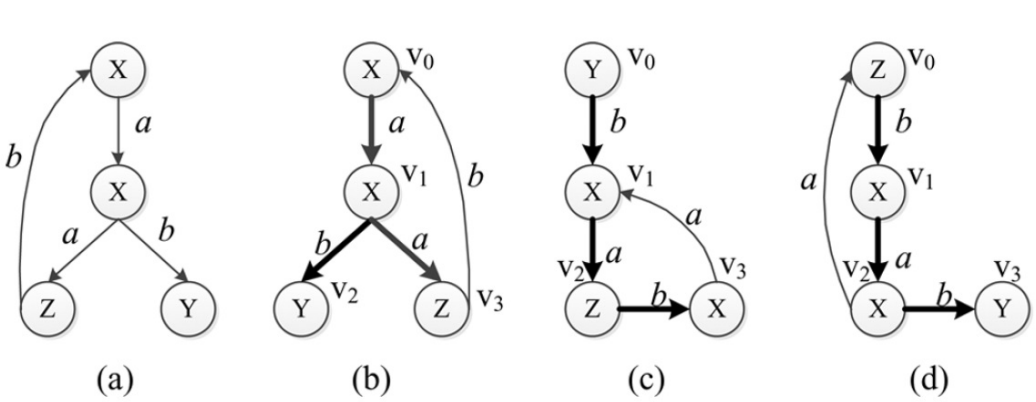


Figure 3.4: Different DFS samples
Source: Li et al. 2014

To compare a pattern to the other, a string edit distance measure SED is applied to their corresponding minDFS code.

2. Common nodes: when comparing two patterns, the more the patterns contain common nodes together the smaller the distance is between them.

3. Backward nodes location: It helps in targeting the more influencing patterns. For example, for a reference model a->b->c->R, where R is the node that needs to be recommender. If the mined patterns are for example a->b, b->c, then the pattern b->c is more influencing as it is closer to the node to be recommended than a->b. This is handled with backward nodes location which in turn will be smaller for b->c than a->b in this example.

Figure 3.5 illustrates the distance measurement formula this recommender's authors implemented. It consists of 3 added variables together, representing the latter factors explained in the last paragraph.

$$\begin{aligned}\mathrm{Dist}(P, P') =& \phi \cdot \mathrm{SED}(P, P') + \varphi \cdot Lo_p(P, P') \\ &+ \psi \cdot (\max_n(P, P') - Co_n(P, P')).\end{aligned}$$

Figure 3.5: Distance measurement equation
Source: Li et al. 2014

3.1.6 Recommendation Based Process Modeling Support: Method and User Experience

This recommender Hornung & Koschmider (2008) is used while the user is modeling the reference model. The user has the chance to use it in two different ways: 1) The first way is through processes querying where the user can invoke a query with certain text about

a certain process. The system will output to them the corresponding recommendations according to the search query. 2) The second way the user can utilize this system is through highlighting the part in the BP model in which they want a recommendation for. For which the system carries out its logic and retrieves to the user the recommendation accordingly.

The recommendations are brought from a shared repository where there are real word Petri net models. The system started off at the initial phase of development with 21 Petri net models. The models existing in this repository are configurable, meaning that business processes models can be added, deleted or altered. The user can specify what type of recommendation they desire. A recommendation can be a process part or the whole business process model.

The implementation logic here is divided into three sections:

1. The first section is the tagging system which utilizes Information Retrieval IR techniques in indexing the BP models. The tagging system indexes each BP model in the repository according to the labels existing in its business processes. The text inside requires some clean up due as it is not unified and rather scattered. The labels are invoked with stop words removal and synonym coverage from WordNet. Then a full TF-IDF calculation is carried out for each BP is calculated for each model. The IDF part is used for reducing the frequent words effect. Usually, it's the less frequent special words in the labels that make things business process parts relevant. Once the BP models in the repository are indexed, the recommender is ready to suggest to the user

2. The second section is the actual recommendation process which the user invokes while working on the reference model. The user inserts as input a search query or highlights a certain process part in the reference model, as explained before. The input goes through the same process of text clean up. Then a mixture of Vector Space Model VSM and the boolean model are used to calculate which of the repository models is close to the input. The VSM uses the TF-IDF values to calculate the angle difference in the vector space between the input vector and the BP model vectors. As for the Boolean model, it handles the cases when inserting queries containing words like (AND, OR or NOT). This is called the Lucene index.

3. The third section is the recommendations ranking. The ranking for the recommendations is based on two principals that are used after the Lucene index calculations. The first one is the rate of errors inserting the new recommendation will do to the reference model. For instance, if the recommendation will cause to have an OR-split is synchronized by an AND-join, then it will have a lesser ranking than another model where this does not occur. The second principal for ranking a recommendation is the popularity of a selected business process. The BP that is selected 5 times for a recommendation from the repository has a higher ranking than BP that was selected 3 times.

3.1.7 Social Software for Modeling Business Processes

This recommendation software Koschmider et al. (2008) is a build up upon the latter recommendation system explained in "Recommendation Based Process Modeling Support: Method and User Experience". That system recommends to users a whole process model or a process part. The recommendation is triggered either by a query interface or a recommendation component. In the query interface, the user queries for recommendations through text insertion. As for the recommendation component, the user highlights the process part in their reference model in which they need a recommendation for. In both cases, the results are retrieved from a shared BP models repository.

The build-up upon the system explained in the latter paragraph takes a different view on the BPM recommendation. It focuses on building a social support network between the users in this system. But the question is, how would a social network help in process recommendation?. The BP models residing in the repository are usually created, added, and modified by users of the system. The use of Social networks can bring about hidden relations and dependencies that can make certain recommendations closer to a user than another. The recommendations in the system at hand are additional recommendations alongside the ones retrieved from enacting the query interface and the recommendation component. These additional recommendations take into account the domain of the searched BP part and the business processes that succeed and preceded it in the reference model.

The social networks mentioned above, are constructed between the users connected by edges. This recommender extracts two types of social networks on these additional recommendations:

1. The first network is a collaboration network in the processes model, where the system extracts the collaborative relations between users in different BP parts. Thus creating a network of connected users. For example, the user chooses a recommendation X then asks the for additional recommendations built upon X. The repository would have, for instance, similar BP parts Y and Z. However, these two BP parts have different users that performed on them. Using the social network constructed, the suitable additional recommendation of either Y or Z will be whoever of them that their users who performed on them are more strongly connected to the performers of X.

2. The second type of social networks utilizes the users' history in the system to find connections between each other. The application of this social network creates clique users that are closely connected according to their history. These cliques are used for propagating changes and notifying the closely neighboured users. The history in the system is represented as a table with users on rows and processes on columns where each cell represents how many times a user i used process j. Distance measurements are calculated, like Minkowski distance, Hamming distance or Pearson's correlation coefficient and the graph is constructed. The cliques form when a threshold is applied

to the edges values that contain the distance between each user. On a side note, Intersecting the user history and process models can bring about another type of clique oriented networks. This is considered also a away of generating these networks. Another way also is through considering the order of the processes.

3.1.8 Advanced social feature in a recommendation system for process modeling

The recommender Koschmider et al. (2009) in hand is an extension over the latter two system explained "Recommendation Based Process Modeling Support: Method and User Experience" and "Social Software for Modeling Business Processes". The way the social networks were used in the latter publication had some drawbacks. After evaluation, the users felt there was not a sort of personal preferencing. The social networks constructed did not take into account what the users explicitly want, as they are automatically constructed. The extension in this publication gives the user the option to add their personal preference to the recommendation propagation. During the evaluation, the users also felt rather isolated. There was not a sufficient changes propagation. This recommender also provides a stronger change propagation functionality by which the users receive proper changes notifications.

The social networks personalization is done through a filtering function where the users insert their network preferences. A metamodel is implemented to make the operation of filtering rather feasible. The metamodel consists of 4 properties representing the user: subject domain, capability, work environment, and individual and group behavior. According to these 4 properties users filter the public social network to get a more personalized output. For example, a filter function with filter(S="government",C="process design",W="IT department") will result a social network containing users that have a subject domain of "government", capabilities of "processes design" and a work environment of "IT department". As for the individual and group behavior, they are the extracted patterns from the user's history in the repository, which is the original social network constructed in the latter publication.

Changes in the repository can be high-level or primitive changes. Any change in a processes model part is labeled part of the high-level changes while changes in nodes is considered primitive. Both can be triggered by an addition, deletion or moving of a process part or a node. These changes can be propagated to users both on the public and the personalised social networks. The propagation is in one of two forms, push and pull services. The first one is a push service, where users belonging to a certain network are systematically notified of a change in their social network. This is helpful to keep a closely related users up to date with any changes they can be concerned with. In addition to the passive push functionality, users can propagate changes in a second form. Users can select certain process models that they want to be notified with whenever a change occurs to them. This is done using the pull service.

High-level and primitive changes are detected with flags and counters to propagate them. Primitive changes occur often, so the user is notified with them if they pass a certain threshold that the user can define or else the user will be spammed with small unworthy primitive changes notifications.

3.1.9 Application of Bayesian Networks to Recommendations in Business Process Modeling

This recommender Bobek et al. (2013) suggests to users business processes and their probability existing in the position they are recommended in. The recommendations position can be backward, backward or auto-completion. The system exploits two main factors for the recommendation to work:

1. The first one is using Bayesian Networks. The system in hand mainly depends on the frequency of the process nodes inside the models and their probabilities of co-existing. One of the approaches famous for handling networks and co-dependant node probabilities inside them is Bayesian Networks. Nevertheless, this system utilizes a repository of models containing BP models that it can use for recommendation.

2. A second active factor that influences this recommender, is the usage of configurable business process models. In short, a configurable business process model is a more generic way of representing a BP model where it's the combination of closely similar BP models. Moreover, they are merged in one final model. Figure 3.6 is an example of a configurable business process combined of 4 business processes. Furthermore, the closely related models are merged together to create one configurable business process model.

The recommendation in general, and Bayesian network probabilities generation in particular, comes about through combining the latter two factors. In details, the system invokes the Bayesian Network theory on the formed configurable business processes residing in the repository. They are consequently transformed into a Bayesian Network. The resulting Bayesian Network in turn is trained to create clusters of which the reference model can be compared to, in order to retrieve the recommendation. The system uses the Maximization Estimation training algorithm for that task. Finally, the recommendation is retrieved to the user in the form of nodes to be suggested throughout the whole reference model and the confidence of their existence in that position. This probability represents the confidence. Moreover, it is derived from the corresponding Configurable Business Process Bayesian Network probabilities values. Figure 3.7 shows the recommendation result invoking this recommender on a BPMN reference model. The red blocks represent the elements that are already existing the reference in the model, while the green ones refer to the new elements that can be included. The values in them indicate their confidence (probability).

This recommender has two points that can be further worked upon according to the authors. The first drawback lies in handling nodes that are not in the repository. When

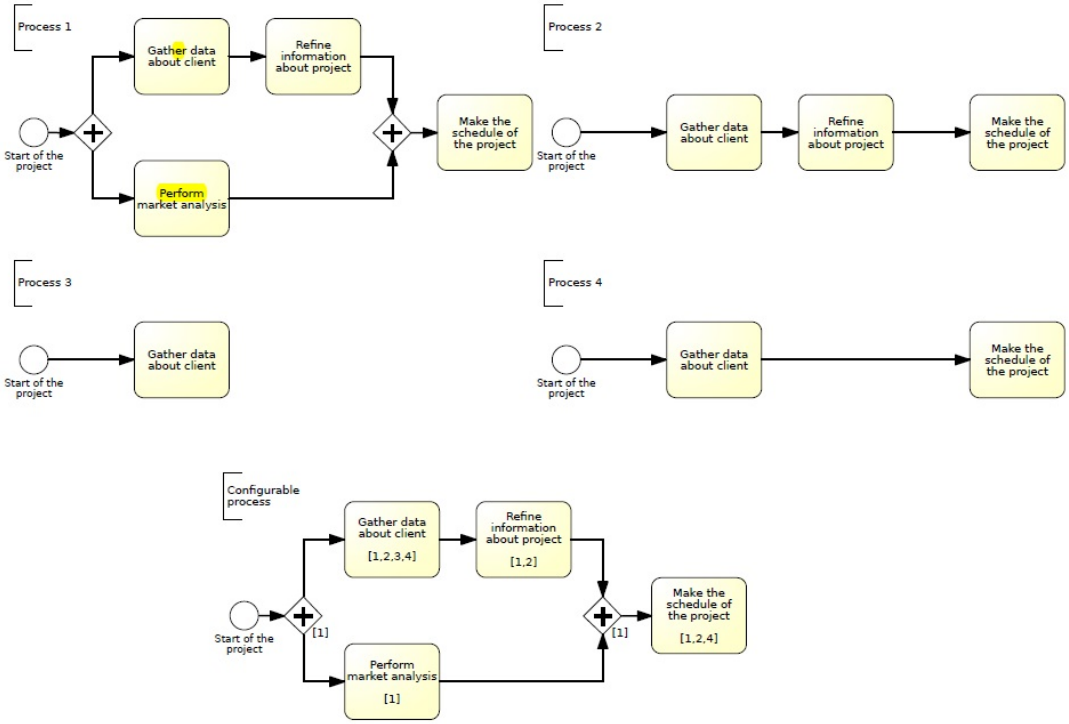


Figure 3.6: Configurable Business Process Model example
Source: Bobek et al. 2013

the user inserts a new process in the reference model that the the trained network has not seen before, the recommendation will crash as the Bayesian probabilistic value will turn to zero thus giving back no recommendation. As for the second drawback, the operation of transforming the configurable process models into Bayesian networks is done by hand thus leaving less automation to the proposed algorithm.

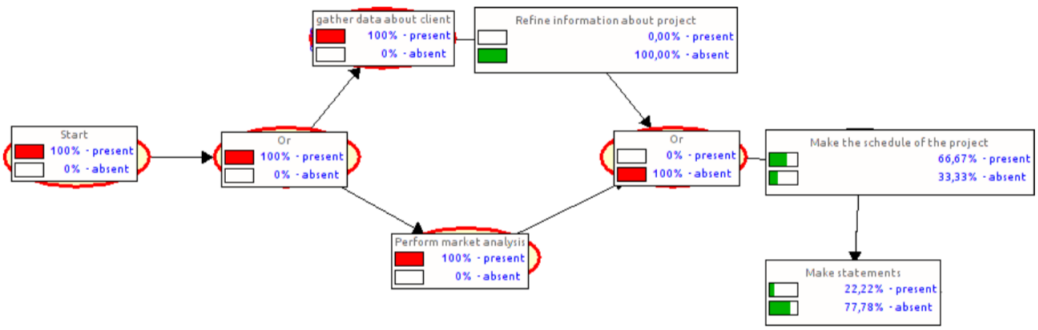


Figure 3.7: Recommendation result
Source: Bobek et al. 2013

3.1.10 Integration of Activity Modeller with Bayesian network based recommender for business processes

This research is a continuation of the work on the recommender proposed in latter paper "Application of Bayesian Networks to Recommendations in Business Process Modeling". The recommender in this paper Bobek et al. (2014), is a structural based recommender that suggests to users forward, backward and performs auto-completion as well. The latter recommender has a repository which contains configurable business processes. These configurable business processes are transformed into Bayesian networks which are trained to be used as for recommending to the users the proper activities.

The recommender proposed in the current paper extends the latter by adding on to it few points to make it more rigid and automated. Due to the fact that one of the drawbacks that the previous recommender has is the lack of automation and flow in it, the new recommender is provided with a platform where modeling and recommendation both can take place. An activity modeller, which is an open source web modeller component called Activiti, is used for this purpose. The recommender is a plugin added to the modeling platform. Moreover, this recommender also tackles the shortcoming that the previous one had in automatically transforming the business process model into a Bayesian network representation. Furthermore, the authors of this paper handled these new features in a new activity recommender architecture. The following figure 3.8 is a representation of the recommender's architecture.

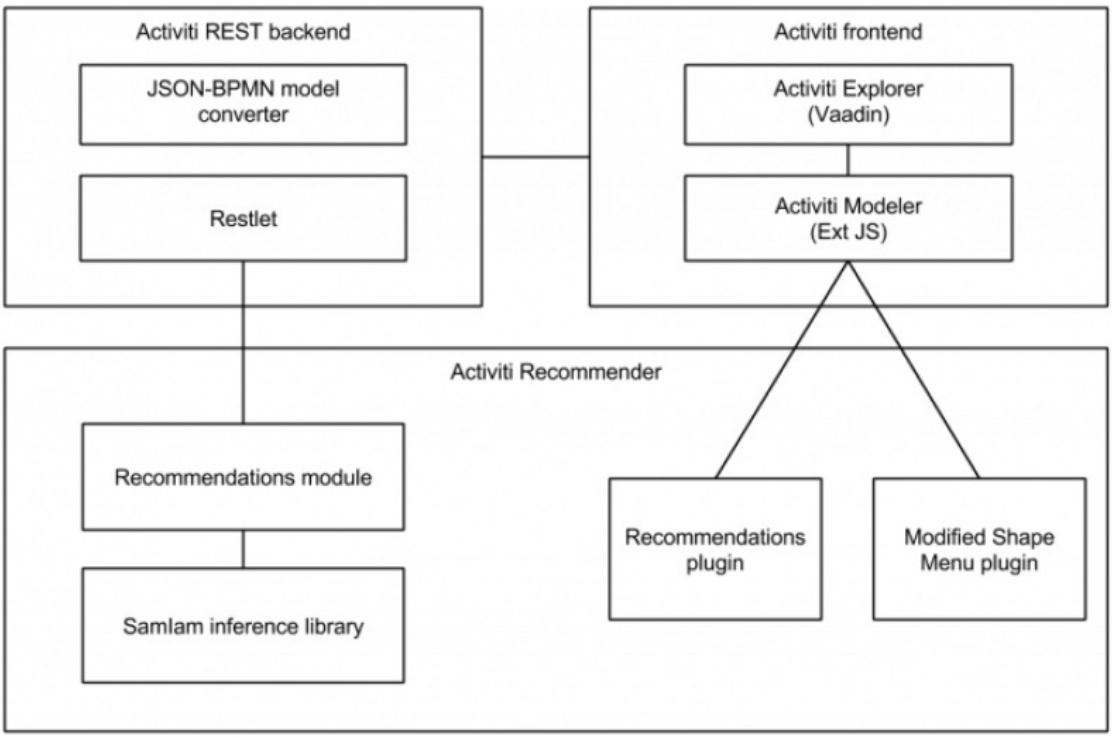


Figure 3.8: The activity recommender's architecture
Source: Bobek et al. 2014

The architecture consists of three main components:

1. Activit frontend: this is the activity modeller web component where the user models the business processes and then require for recommendations.

2. Activiti REST backend: this component transforms the BPMN created in the front end into a Bayesian network then trains it against the configurable business processes in the repository using probabilistic machine learning algorithm Maximization Expectation.

3. Activiti Recommender: the final component uses the trained Bayesian network and traverses it to retrieve the recommendations to the user. Furthermore, in order to match the process under design (the reference model) and the trained Bayesian network, a text based matching is used. The matching mechanism helps associate the activities in the reference model with their corresponding targets in the Bayesian network in order to retrieve an appropriate recommendation. The textual matching is performed using a Levenshtein distance algorithm on the activities' labels.

Figure 3.9 shows the proposed extended recommender over the Acitivi modeler where it is extended with a user-interface to handle the recommendation calls. The user-interface is represented in the recommendations menu and the action options over the elements. For example, the action "First recommended element - Click or drag".

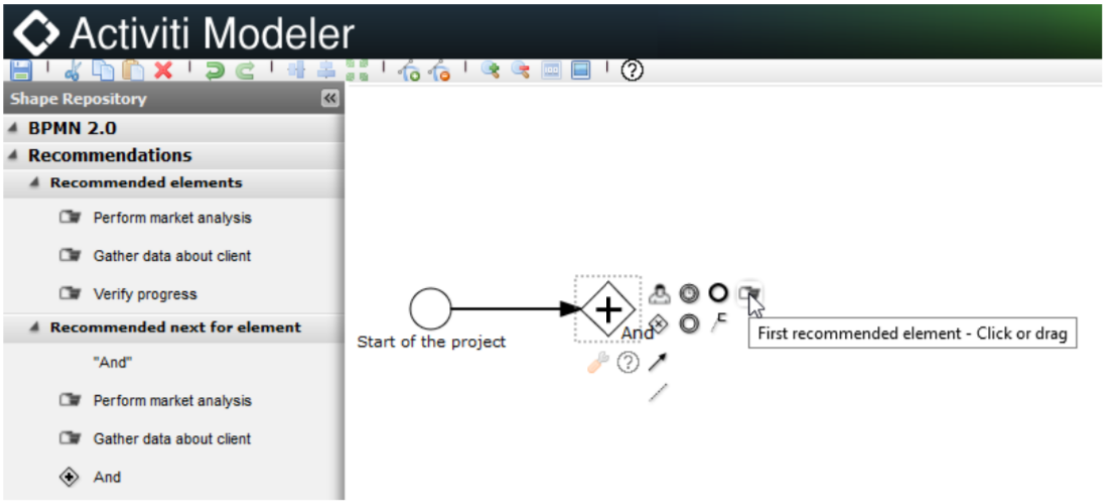


Figure 3.9: The activity recommender
Source: Bobek et al. 2014

3.1.11 Context-Sensitive Textual Recommendations for Incomplete Process Model Elements

This recommender Pittke et al. (2015) handles incompleteness in business process models. A business process model can contain incompleteness when contextually compared to a conceptual model. Moreover, this incompleteness is categorized as either structural or textual. A structural incompleteness is when an activity or an event is missing, thus contextually violating the structure. On the other hand, the textual incompleteness in a model's node is represented in the lack of textual information provided in the label,

thus giving a shortage of context to the node. The recommender proposed in this paper returns naming recommendations for the textually incomplete nodes in a business process model.

A node is said to be incomplete if it misses sufficient information describing it. A node should always contain an action and a business object. For example, in the label "check password", the action is "check" while the business object is "password". Any label missing one of these two is considered incomplete. After defining the textual incompleteness, the recommendation phase come which is broken into two consecutive stages: recommendations creation and recommendations ranking.

1. Recommendation Creation: Not much is mentioned to how the contextual correlation is performed to retrieve to a certain node. However, the paper mentions three context layer strategies through which recommendations can be created:

    a) Local Process Model: this strategy uses all actions and business objects residing in the reference model as possible recommendations.

    b) Process Model Collection: as for the strategy it considers all Business process models in a given repository.

    c) External Corpora: the last strategy utilizes certain text corpora to find terms that co-occur with the missing text in the node.

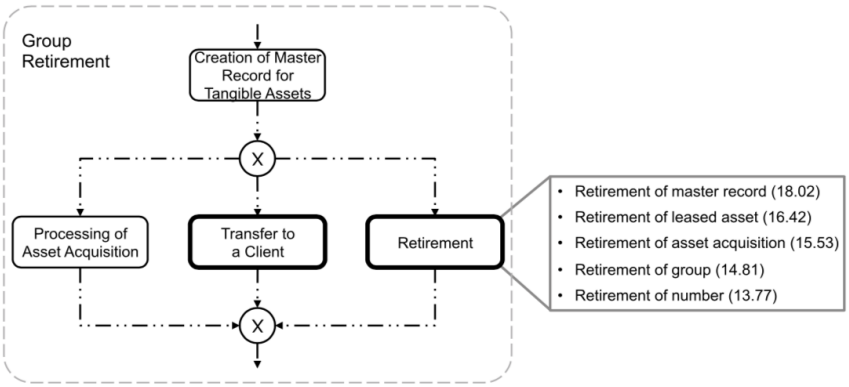
2. Recommendation Ranking: after creating a certain recommendation list for an incomplete node. The recommendations are ranked to retrieve the closest one in context. The ranking procedure uses sense clusters to group the BP model nodes together. Each sense cluster is a group of closely related terms grouped together. The ranking occurs through creating sense clusters for the labels of the reference model nodes. Followed by conducting a similarity measurement between each recommendation in the recommendations list and the sense clusters created. Finally, the list is sorted descending. This causes the recommendations with the highest similarity score come first.

This approach is tested through an evaluation experiment where two different BPM test collections containing both in total 1695 BP models which had around 782 incomplete nodes. The experiment was performed on a small sample on which a group of 5 users (2 modeling novices and 3 modeling experts) decided upon the usefulness of the recommendations given by this recommender. The users rated a certain recommendation on a 4-point-Likert-scale from very useless (-2) to very useful (+2). The ranking calculation was performed on three different conditions: top 1, top 5 and top 10 recommendations. Moreover, the evaluation found out the 53% of the recommendations were useful. They also indicated that whenever more recommendations are considered, meaning top 1, top 5 and top 10, the higher the recall values are and the less the precision is. This means that the top 1 recommendation is more likely to be more precise than the top 5 according to users rating. However, the top 5 will be able to give more information to the user than the top 1. Figure 3.10 describes the recall and precision values for the top 1, 5 and 10

recommendations. Lastly, figure 3.11 is an example of top 5 textual recommendations for a BP model called “Grouped retirement”.

	Top 1	**Top 5**	**Top 10**
No. Relevant & Retrieved	23	86	149
No. Non-relevant & Retrieved	8	66	133
No. Relevant & Not retrieved	126	63	0
Precision	0.74	0.57	0.53
Recall	0.15	0.58	1

Figure 3.10: Top 1, 5 and 10 recommendations Precision and Recall values
Source: Pittke et al. (2015)



Activity	**Top 5 Recommendations**	**Ranking**
Transfer to a Client	Transfer Time Specifications to a Client	25.76
	Transfer Balance Sheet Items to a Client	19.51
	Transfer Planned Sales Quantities to a Client	18.87
	Transfer Group to a Client	18.26
	Transfer Personnel Costs to a Client	16.24
Retirement	Retirement of Master Record	18.02
	Retirement of Leased Asset	16.42
	Retirement of Asset Acquisition	15.53
	Retirement of Group	14.81
	Retirement of Number	13.77

Figure 3.11: Group Retirement model. Top 5 Textual Recommendations for the activity Retirement
Source: Pittke et al. (2015)

3.1.12 On the Automatic Labeling of Process Models

The recommender in this paper Leopold et al. (2011) supports users by automatically providing a label for a business process model fragment in particular or a business process model in general. This type of recommendation is considered a textual recommendation where only the label of a certain element or business process model is given a recommended value. Moreover, textual recommendations are beneficial for the clarity of a business

process model. Ambiguous, unclear labels can be misleading, thus making the model counter-intuitive. Furthermore, the contribution provided in this paper provides a stronger focus on how to extract a recommended label from a business process model. The research does not make difference between providing this recommended label for a whole business process model or a single activity in it.

On a similar context, many researches in the field of textual recommendation use the notion of verb-object style in modeling an activity label where the verb represents the action while the object represents the business object. For example, in an activity label "Activate Material Price", the verb "Activate" is the action which the corresponding activity performs, while "Material Price" is the business object on which the action is performed. Moreover, the verb-object style is used in this paper to model the labels in a business process as well as automatically producing a recommended label. In details, generating the mentioned textual recommendation goes through 3 phases. During the first phase, the recommender parses all textual information for each activity and modeling them into a verb-object notation. Afterwards, the gathered information are passed to a classifier that produces a resulting label from the given input. This classifier, which is the core of this recommender, can be done in 5 strategies. They will be described in details in the next paragraph. As for the last phase, the produced result generated in the second phase is transformed into the verb-object style and is produced to the user.

The process of classifying a label to be recommended can be carried out with 5 strategies:

1. Dominating element strategy: A dominating element is a textual action or business object that exists the most in a business process model. In other words, the most frequent action or business object is considered a dominating element. Consequently, a subordinate elements set extracted containing all the actions and business processes associated with the dominating element. This set is vital in the next strategy.

2. Conjunction of activities: the subordinate elements set is used to compliment the missing part next to the dominating element by conjuncting them together, whether an action or a business object. This is carried out through two different approaches. Firstly is Lixical conjunction, where the subordinate elements are lexical-classified in a hypernym lexical database like WordNet to produce a new introduced element. Eventually, the lexically introduced element is conjuncted with the dominating element to retrieve to the user as the final recommendation. As for the second approach of conjunction, it's using Logical conjunction. Logical conjunction in simple form is conjuncting all subordinate elements together to the dominating element and retrieving the result to the user as the final recommended label.

3. Start or End Events strategy: a start or end event in a business process carry valuable context information for the model. Therefor, one of the proposed strategies utilizes the start and end events, Moreover, the start events are classified by labels that contain verbs in the present. An example for that is, "Asset is to be created", while on the other hand an activity label like "Asset was created" is considered an end event.

4. Main Activity strategy: each business process contains a main activity which is vital for the model itself. An activity that is central of importance. The authors propose that most probably the main activity resides at the beginning or at the end of the business process model. Nevertheless, the label of the main activity is used as a proposal for the new recommended labels.

5. Semantic Naming strategy: This approach uses the broader context of the business process model, contrary to the latter strategies that use an activity in particular whether dominating and conjunction element, main activity or event driven ones. In details, the context is driven from relationships between activities and their labels through setting semantic rules that can be used to generate a new label. An example for that is a business process that contains the two events "activities Delivery for Returns" and "Goods Receipt Processing for Returns". In semantic logic, the term "shipping" can be considered as a broader representation of the two events. Thus it could be used as a proposal for the new label. This approach requires external ontology libraries.

3.1.13 On the refactoring of activity labels in business process models

This paper Leopold et al. (2012) introduces a service for refactoring a business process model activity label. This service can be used in modeling tools as a form of textual recommendation for users while modeling. Furthermore, an activity label can vary in structure and style, yet most of the time it contains an Action and a Business Object. As explained previously, the Action represents the actual type of verbal action which the activity executes. On the other hand, the Business Object is what the Action is performed on. Business process model activities labels may carry the same meaning, but be available in different structures. Take an example of an activity label with the Action "Create" and the Business Object "Invoice". It may come in the style of "Create Invoice", "Invoice Creation", "Creation of Invoice" and other forms. Therefore, this recommender classifies the label given to it to one of seven label styles. These styles will be explained in details in the next paragraph. Finally, the recommender extracts from label the Action and Business Object according to the classified label style and recommends to the user a new label using the Verb-Object approach where the Verb represents the Action while the Object represents the Business Object. The authors to this recommender absorbed the sentences in the labels and managed to categorize them into 7 different styles. The following figure 3.12 show the aforementioned styles.

In details, these styles are as follows:

1. Verb-Object: Example to that is "Create Invoice". This style is what the authors try to make all other labels styles eventually fit.

2. Action-Noun (np): Example to that is "Invoice Creation".

3. Action-Noun (of): Example to that is "Creation of Invoice".

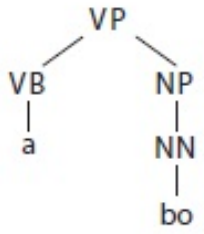
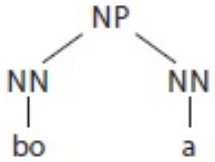
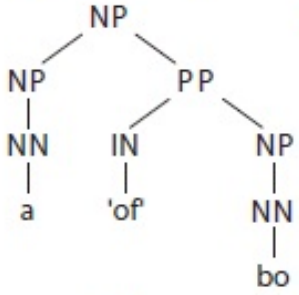
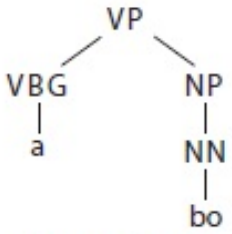
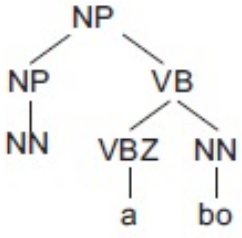
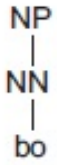
Labeling style	Structure	Example
Verb-object VO		Create invoice
Action-noun AN (np)		Invoice creation
Action-noun AN (of)		Creation of invoice
Action-noun AN (ing)		Creating invoice
Action-noun AN (irregular)	⟨*anomalous*⟩	LIFO: valuation: pool level
Descriptive DES		Clerk answers call
No-action NA		Error

Figure 3.12: Labels Styles
Source: Leopold et al. 2012

4. Action-Noun (ing): Example to that is “Creation of Invoice”.

5. Action-Noun (Irregular): This style handles the sentences where irregular characters exist. For instance the label, “LIFO: valuation: pool level” contains an irregular character which is “:”.

6. Descriptive: This style handles the sentences that are too long that don’t contain an actual Action-Noun structure. An example for that is “Accounting creates invoice and Applies for registration”.

7. No-Action: This style handles labels that contain no action. An example to that is “error”.

Moreover, the label style classification operation plays an essential role in extracting the Action and Business Object from the label according to the style, thus refactoring the model labels. But it’s not the only step to achieve that. Additionally there are other steps that are applied sequentially until the activities in the model are refactored. These phases are as follows:

1. The label style classification phase: Classifying the style of the label is not a straightforward operation due to the fact that many labels are complex. Namely, in order to classify a label, it is checked on 4 different levels analysis. If the label is not classified in the first level analysis, then it’s taken to the further level and so on. Firstly, a) Label analysis level. In the first level, the recommender tries to assign each activity label to a style. This is done through using the activity label itself and checking its structure whether its Verb-Object, Action-Noun or Description. However, some activities contain labels that are not directly classified. These unclassified activities are in term passed to the second level, b) Model analysis level. In this level, the unclassified labels from the previous phase are checked with the other model’s remaining activities labels. If an unclassified label has its first word occurring in other model labels then, the label is assigned to the class of the other model labels. After this phase, the remaining unclassified that were not classified in the level b, then are passed onto the next level c) Model collection analysis level looks through all over the repository of all the models and tries to classify the remaining labels. Lastly, d) Natural Analysis language level assigns all the remaining unclassified labels that aren’t matched in the latter three levels. It utilizes a natural language lexical database to classify whether a label is Verb-Object or Action-Noun. This is done by checking the probability of the first word in the label if it’s a verb or a noun.

2. Analysis of Action-Noun labels phase: Moreover, the last phase categorises the business process model labels into Verb-Object, Action-Noun and Description styles. During this phase, further analysis are performed to handle the different options for Noun-Action styles. This is beneficial in extracting the Action and Business Object in the next phase.

3. Derivation of the Action and Business Object: This phase is a direct application of Action and Business Object extraction techniques according to the label itself.

For example, in a Action-Noun (ing) like “Invoice Creating”, it is easily identifiable from the structure. “Creating” is the Action and “Invoice” is the Business Object. For more details, “Creating” is transformed into its infinitive form, thus becoming “Create”

4. Composition of a Verb-Object output activity label: Finally, the Action and Business Object are combined together and giving to the user as the recommended refactoring label value.

Finally, the authors evaluated the results of this recommender over experiments during which they utilized 3 different test collections. The first one is a SAP reference model using the EPC notation. The second one is a TelCo collection containing ADONIS models. The last collection is Signavio Process model collection in BPMN. Moreover, main core of the experiment was to assess the 4 phases this recommender performs to produce the recommendations. The 4 phases are judged with precision and recall and F-measure check for each test collection (SAP, TelCo and Signavio). The evaluation showed strong results. The percision, recall and F-measure values were relatively high. In addition, the evaluation experiments included an experiment that compares the result of the recommendation results to the Standford Parser. The final evaluation indicated that the recommender is relatively better than the Standford Parser in refactoring the models' labels. This is believed due to the broader purpose of the Standford Parser and that this recommender is tailored for the task of refactoring BP models' labels.

3.1.14 Supporting flexible processes through recommendations based on history

In the new era of technology advancement, Process Aware Information Systems PAIS came about to keep the process models up with the changing environment. Business Process models are required to be responsive to changes. For example, profit maximization or cost reduction. This recommender, contrary to other recommenders explained reviewed in this chapter, doesn't suggest new activities or connections. However, it suggests to the users helpful information on their reference model activities that fit their target better. In other words, it shows target related descriptive information on each concerned activity in the reference model. Moreover, the target can be in the form of time reduction, cost reduction or profit maximization. Furthermore, this recommender Schonenberg et al. (2008) is invoked on PAISs. These PAISs vary from unfinished business process model or a process model that needs enhancement. Moreover, the recommender uses the event logs of these PAISs and recommend to the user next activities in the model that would help reach the user's target goal.

The events logs are updated for each time a business process is used by a model's log trace. Scaling this on a larger scope results in a collection of business process event logs (log traces) containing models activities execution paths. Furthermore, the constantly changing paths preserved in event logs can be exploited to enhance the current state of

the business model as they hold valuable information and patterns. These paths can be utilized by comparing the reference model to them and extracting a recommendation afterwards. For this sake, the recommender requires the user to send an input containing two requirements 1) partial trace and 2) the enabled activities. The first requirement is the partial trace. The partial trace is a path of activities connections representing the history of the reference model's activities execution. As for the second requirement, the enabled activities indicate the activities in the reference model that are used in the time when the time when the reference model requires the recommendation. Their purpose is to ensure that no violation of constraints occur. In summary, this recommender takes the two latter mentioned request inputs and extracts to the user recommendations fitting their goal accordingly.

Abstractly, the recommender retrieves the recommendations that fit target goal fitting by taking each enabled activity, checking its effect on through out the event logs. The effect is measured by two calculations: the Do and the Don't. The Do calculations measure the cost throughout the event logs if the enabled activity is used, while Dont calculations are for the remaining enabled activities. This approach is explained in the example shown in figure 3.13.

The example represents a reference model with a partial trace $< D, F >$ and enabled activities (A,B,C). Where the goal is to minimize the cost. The table elements are:

- Log: represents the events logs, and the cost values are inquired from each logs value. Meaning that they are given to the recommender, not calculated.
- w: represents the weight for each log trace in the events logs, given the partial trace $< D, F >$ of the reference model. For example, ABC has a weight of 0 because it contains neither of the partial trace elements, while DBC has a weight of 0.5 as it contains the activity D.
- s: indicates the support of each enabled activity. In other words, the existence of each enabled activity in each events logs log trace.
- Do calculations: To calculate the Do calculations for the enabled activity A for example, the events logs are traversed for each log trace that contains the activity A. In this case they are (ABC, DFA, CCA). Then the weights are multiplied by the costs for these log traces and divided by their weights sum. The Do value for A given the partial trace $< D, F >$ is 1000.
- Dont calculations: Carrying on with the same enabled activity A. The Dont calculations are concerned with the log traces that don't contain A but contain the remaining enabled activities B and C. Therefore, the resulting traces for the log traces that contain B and not contain A are (DBC, FBC, DFB). As for the log traces that contain C and not contain A are (DBC, FBC, DFC). Same weight cost multiplication is performed. The resulting Dont values for A given the partial trace $< D, F >$ is 1125.

The Do calculations indicate the cost if the activity is introduced while the Dont indicate the cost if it weren't there. The difference between the two values is the recommendation to the user. Continuing with the same example in figure 3.13, the user goal is to minimize the cost. Therefore the recommendation will be as follows:

1. B: (1000 - 1500 = -500) Least cost
2. A: (1000 - 1125 = -125)
3. C: (1250 - 1250 = 0) Most cost

Log		Weight and support			
σ	cost	$\omega_s(\rho,\sigma)$	$s_s(\rho,\sigma,e)$		
			$e=A$	$e=B$	$e=C$
ABC	900	0	⊤	⊤	⊤
DBC	500	0.5		⊤	⊤
FBC	500	0.5		⊤	⊤
DFA	1000	1	⊤		
DFB	1500	1		⊤	
DFC	2000	1			⊤
DFH	1260	1			
CCA	1680	0	⊤		⊤

$$do(A,\langle D,F\rangle,L) = \frac{0\cdot 900+1\cdot 1000+0\cdot 1680}{0+1+0} = 1000$$

$$do(B,\langle D,F\rangle,L) = \frac{0\cdot 900+0.5\cdot 500+0.5\cdot 500+1\cdot 1500}{0+0.5+0.5+1} = 1000$$

$$do(C,\langle D,F\rangle,L) = \frac{0\cdot 900+0.5\cdot 500+0.5\cdot 500+1\cdot 2000+0\cdot 1680}{0+0.5+0.5+1+0} = 1250$$

$$dont(A,\langle D,F\rangle,L) = \frac{(0.5\cdot 500+0.5\cdot 500+1\cdot 1500)}{(0.5+0.5+1)+(0.5+0.5+1)} + \frac{(0.5\cdot 500+0.5\cdot 500+1\cdot 2000)}{(0.5+0.5+1)+(0.5+0.5+1)} = 1125$$

$$dont(B,\langle D,F\rangle,L) = \frac{(1\cdot 1000+0\cdot 1680)+(1\cdot 2000+0\cdot 1680)}{(1+0.5)+(1+0)} = 1500$$

$$dont(C,\langle D,F\rangle,L) = \frac{(1\cdot 1000)+(1\cdot 1500)}{(1)+(1)} = 1250$$

Figure 3.13: Algorithm Calculations for BP With enabled activities recommendation
Source: Schonenberg et al. 2008

3.1.15 Business Process Models as a Showcase for Syntax-Based Assistance in Diagram Editors

Diagram editors are mainly alike to each other in terms of abstract similarity. But extensions over them are being applied to be make them applicable for certain service or to solve

a certain problem. These extensions can come in the shape of meta tools, enriching the diagrams with additional information to be utilized for specific purposes. The recommender proposed in this paper Mazanek & Minas (2009) is based on this idea. Moreover, the meta information here uses a grammar language, derived from the model under development in the diagram editor. These information are later on used to extract relations from the reference models to be used to assist the user. The assistance comes in 4 forms:

1. Auto-completion: Incomplete BPM diagrams come as a result of an intermediate diagram or an experienced user that created a missing link. The auto-completion functionality allows the user to fill the missing link with an activity or more. In addition, the user sets the size of nodes to be completed. Eventually, the user will be given all different cases of completion scenarios, which he can choose from.

2. Example generation: The user can receive full examples based on the connections extracted. After Which the user can choose of which he can benefit from. These examples can be of the size the user sets.

3. Editing operations: It is possible for the user to select certain activities and impose editing operations on them according to their context. Furthermore, these operations can be adding to the selected component or removing it. The different options are presented to the user and he chooses the appropriate accordingly. In details, the different options are drafted according to the connections extracted.

4. Auto linking: When detecting an incomplete BPM diagram, one of the actions that can be performed on it is auto-completion which is explained here. The second option is auto-linking. Auto-linking is closing the gap in the disconnected nodes, thus shutting the missing link.

The operation of connections extraction from the reference mode, is done by using a grammar language based on an editing diagram tool. The research uses the DiaGen framework, with an integrated recommendation service. This recommendation service implements a graph grammar to model the meta information needed to be extracted. In short, the reference model goes through certain stages in order to develop a certain grammar over which the user assistance is brought about. During one of the stages, reference model is transformed into an Abstract Syntax Graph ASG representation. The following two figures 3.14, 3.15 show an example of a “Sales Department” in simple BPM and its ASG representation, accordingly.

The ASG representation is vital for setting the rules of the graph grammar. These rules, which are called hypergraph grammars, describe each connection in the model. Moreover, the rules are extracted on a sequential approach. The hypergraph grammar consists of non-terminal and terminal hyperedge used together in sequentially deduced rules called productions. Figure 3.16 shows the hypergraph grammar productions extracted from the “Sales Department” ASG.

Nonterminal hyperedges, presented by the light green activities in the figure, represent the context-free edges. In one way or another, it’s an abstraction of a certain edge. On the

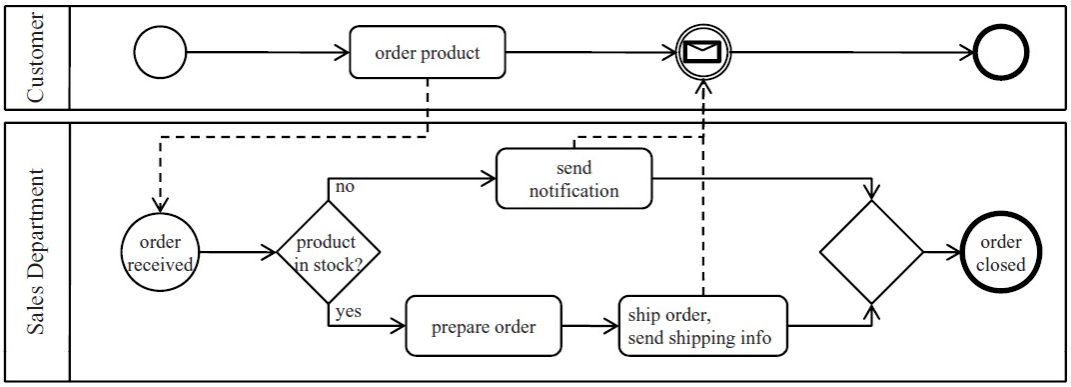


Figure 3.14: Sales Department BPMN Representation
Source: Mazanek et Minas. 2009

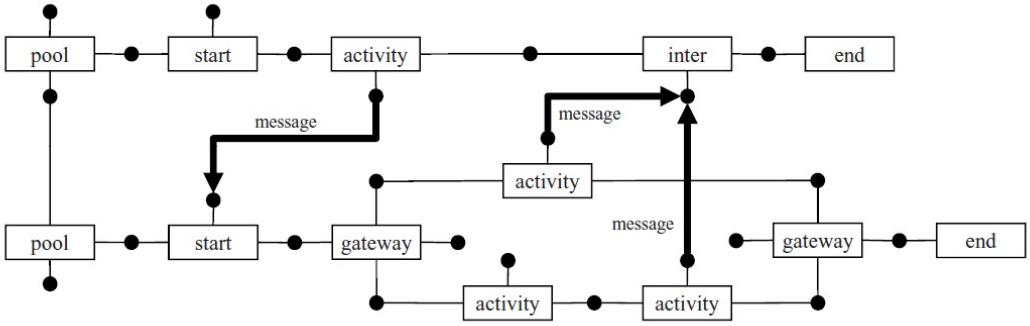


Figure 3.15: Sales Department ASG Representation
Source: Mazanek et Minas. 2009

other side, terminal hyperedges, the white activities, are terminally existent in the actual ASG.

The productions introduced in the figure, are the rules of which the 4 forms of assistance is brought about. In other words, the recommender uses these productions to produce recommendations to users in the form of (Auto-completion, Example generation, Editing operations or Auto-linking). The following will be an example of how to use productions to produce assistance in the form of Example generation:

1. P4 is used, thus the current suggestion diagram is start->Flow->end. For distinction purposes, non-terminal hyperedges start with capital letters while terminal ones start with small letters.

2. P6 is used afterwards to determine the “Flow” non terminal hyperedge, making the current suggested flow start->FlElem->end.

3. P8 is chosen after that to determine the “FlElem” non terminal hyperedge, and is eventually replaces with the terminal hyperedge “activity”.

4. Finally, the diagram to recommend is start->activity->end.

The user receives multiple diverse shapes of recommended diagrams each depending on the size needed for recommendation and the production used for it.

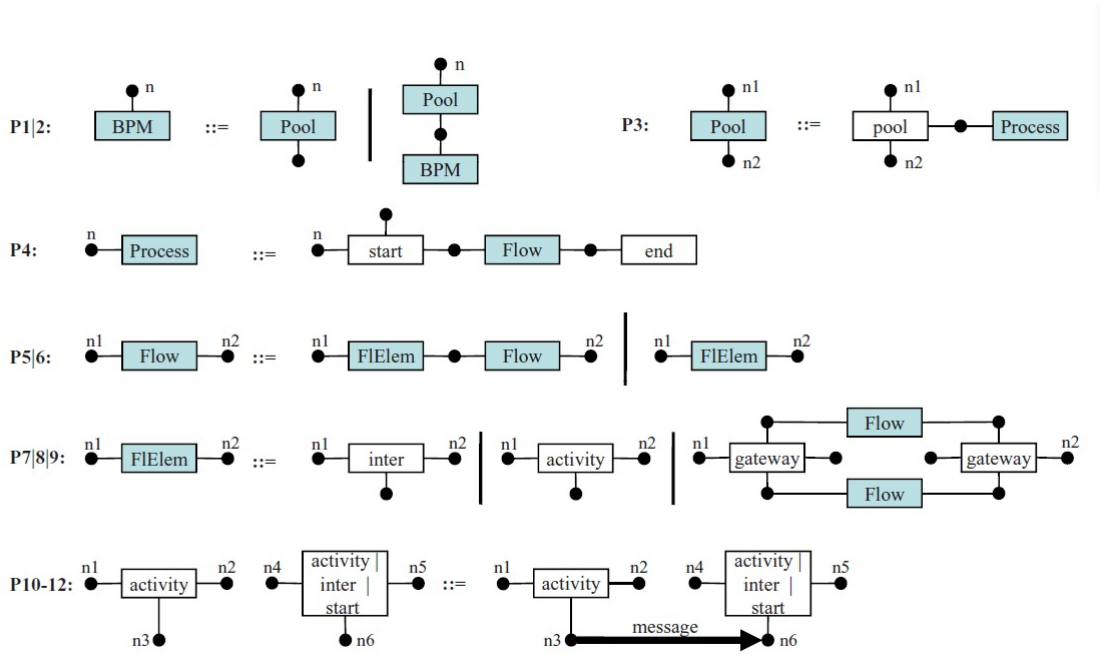


Figure 3.16: Sales Department extracted hypergraph grammar productions
Source: Mazanek et Minas. 2009

3.1.16 Towards Auto-Suggested Process Modeling – Prototypical Development of an Auto-Suggest Component for Process Modeling Tools

This recommender Shitkova et al. (2013) suggests to users what activity comes next in the workflow. It mainly functions on the reuse of previously used business process models residing knowledge databases. The implementation takes an architectural point of view that isolates the recommendation service from the different modeling environments and notations. Moreover, the recommender's architecture selection will be explained in the next paragraph. On a similar topic, there are several business process languages like BPMN, EPC and BPEL. Therefore, the authors of this research wanted to create a notation independent recommender system to broaden the knowledge database. This way, if a user modeling on a BPMN environment may receive a suggestion coming from an information, which is residing in the knowledge database, taken from an EPC model.

The recommender's self contained language independent architecture, is based on the usage of activities labels and connections. The authors proposed an interface between the recommender and the modeling environment that transforms the business process model from its modeling notation format to a native format, using JSON, that is stored in the knowledge database. Furthermore, the knowledge database, existing in the recommender, is used directly for retrieving the suggestions to the user. Extending on this matter, deciding what model part to suggest goes through two phases:

1. Deciding which are the potential suggestions: Furthermore, the potential recommendations are characterised according to how frequent the sequence appear in which

the predecessor is similar to the last node in the reference model. For example, when the user is modeling the business process model A->B->C, and on the same time the knowledge database contains a sequence C->E that occurs multiple times, then the element E is considered a potential suggestion. To check and compare element's labels, Levenshtein string distance measurement is utilized. Furthermore, another factor that affects the potentiality of a suggestion is the date of usage. An element that is used lately is most probably to be used again, thus the author proposed a formula to calculate the score for each element. The following figure 3.17 presents the formula mentioned. The h represents the frequency of usage while a represents the days since the element was last used. It other words, the formula interprets that an element that is one year old will lose half of its score.

$$S(h,a) = h * e^{-\frac{\log 2}{365} * a}$$

Figure 3.17: The Recommender's formula
Source: Shitkova et al. 2013

2. Deciding which one of the potential suggestions to give back to the user: the authors proposed two approaches. The first one implements a threshold over the score. In details, any element having a score, obtained by the score function explained, less than a certain threshold, will not be taken. However, this approach is not good for usability purpose. For example, given a case where many elements pass the threshold, then the user will be provided with excess supply of suggestions. Therefore, the second approach suggests on a taking a fixed number of suggestions without care about the score. Even though this method can provide the user with some unfitting suggestions, but at least it will not spam him/her with suggestions.

Above the latter concepts, the recommender has a web interface that facilitates the interaction between the recommender and the modeling tool. It handles the suggestions in terms of communicating and interfacing them, where the models are transferred from a notation to a native one and vice versa. In addition, it shows the user the suggestions and applies their effect on the reference model being modeled. Furthermore, figure 3.18 elaborates how the recommender flow starts from invoking the model till enacting the suggestion. The flows move through the following stages:

1. Initially the user selects the models he/she needs to work on.
2. The modeling tool invokes the modeling tool interface and passes on the model information on to it.
3. JSON-based communication occurs here.
4. The recommender's web interface then transfers the JSON object sent to the native notation used in the knowledge database.
5. Querying the database takes place next, to the retrieve the possible successors to the last element being developed in the reference model.

6. After that, the result from the database querying arrives containing all the possible suggestions.
7. The recommender starts to call the score function on the possible suggestions to calculate their score values. Eventually, the result is passed on via the web interface.
8. this stage, communication back to the modeling tool interface takes place with the suggestion.
9. Afterwards, the transferred suggestions are transformed back to the modeling tool language notation through the modeling tool interface.
10. The suggestions are shown to the user here.
11. The user inserts his/her input whether to accept or reject the suggestions.
12. If the user takes the suggestion, then in this step it's inserted in the reference model.
13. The user is shown the resulting process.
14. The user see the resulting process here.
15. 16,17,18 The recommender's knowledge database must be informed of the change occurred in the reference model in order to update the element's frequency and last time used index.

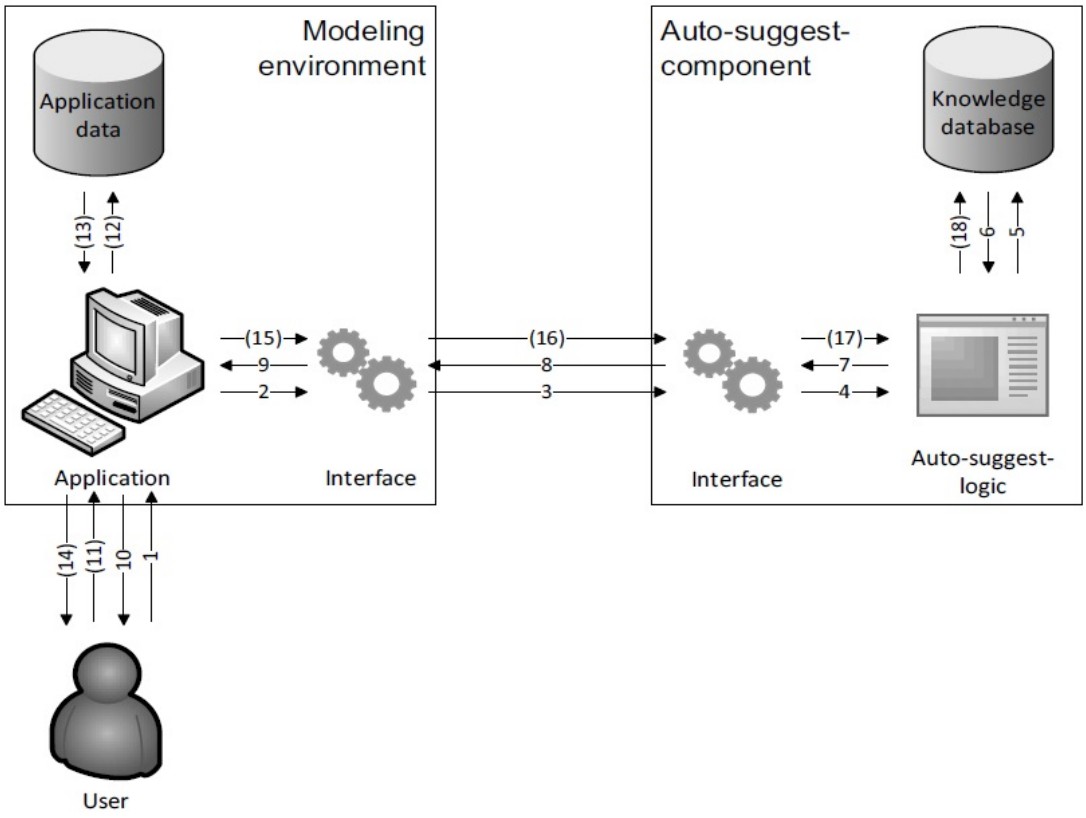


Figure 3.18: The Recommender's flow
Source: Shitkova et al. 2013

3.1.17 Automatic user support for business process modeling

The recommender proposed in this research Betz et al. (2006) structurally suggests to users what activities to come in their reference model. Generally speaking, this recommender is based on Semantic Business Process Modeling. In details, Semantic Business Process Models combine both business process modeling, in addition to semantic ontologies reasoning. The introduction of semantic reasoning helps automate the processing operation of the models. It helps in understanding the context of the process models, thus detecting similarities between one another. Moreover, the similarity calculation is done between the reference model, with stored process templates. The process templates closest to the reference model are used for recommendation. Not only is the recommender's functionality based on Semantic based context similarity checking but also on business process validation for structural errors.

The main functionality this recommender depends on is context matching based on semantic reasoning. The authors implemented an approach where they translate Petri net models, a business process management languages, to Web Ontology Logic (OWL) representation. In short, OWL is an ambiguous format that provide reasoning over models ontologically. Moreover, the authors utilized a high-level variant of Petri nets called Pedicate/Transaction Pr/T nets, where predicates represent places, while Transactions represent a logical expression of which the predicates occur. Each model is transformed

from Petri nets to OWL representation, where place is represented by a concept Place, transactions by Transactions and arcs by FromPlace and ToPlace. The following figure 3.19 explains this more in details.

$$
\begin{array}{ll}
PetriNet \equiv & \geq 1 hasNode.(Transition \sqcap Place) \\
 & \sqcap \geq 1 hasArc.(FromPlace \sqcup ToPlace) \\
Transition \equiv & placeRef.Place \sqcap = 1 haslogicalConcept.LogicalConcept \\
Place \equiv & transRef.Transition \sqcap = 1 hasMarking.IndividualDataItem \\
FromPlace \equiv & \geq 1 hasInscription.Delete \sqcap \exists hasNode.Place \\
ToPlace \equiv & \geq 1 hasInscription.Insert \sqcap \exists hasNode.Transition \\
LogicalConcept \equiv & = 1 hasConditon.Condition \sqcup = 1 has.Operation.Operation \sqcap \\
 & \exists hasAttribute.IndividualDataItem \\
IndividualDataItem \equiv & \geq 1 hasAttribute.Attribute \\
Delete \equiv & \forall hasAttribute.IndividualDataItem \\
Insert \equiv & \exists hasAttribute.IndividualDataItem \\
Atrribute \equiv & \leq 1 hasValue.Value \\
Value \equiv & hasRef.Value \\
Condition \equiv & forall(string) \sqcup exists(string) \sqcup and(string) \\
Operation \equiv & function(string)
\end{array}
$$

Figure 3.19: Petri net Translation to OWL ONtology
Source: Betz et al. 2006

Building on the latter information, the similarity measurement is carried out on the labels in the OWL transformed representations of the models. This operation takes places with regards to three factors:

1. Syntactic factors: The authors proposed using the Levenshtein distance measure to compare two labels syntactic similarity.

2. Linguistic factors: Syntactical measurements are not always sufficient for measuring similarity. Thus using a linguistic library like WordNet helps close the gap provided by syntactically different labels yet linguistically close.

3. Structural factors: According to the Tr/P nets to OWL representation shown in the latter figure, each Petri net place it contains features. These features are Place, Attributes, Values and Transactions. Furthermore, the similarity check based, whether syntactic or linguistic, takes place according to the type of feature. Figure 3.20 shows the different options and each option's similarity check choice and weight.

In addition to the previously explained feature of semantic context similarity check, this recommender detects violations that the new suggested addition may add to the reference model being developed. It covers deadlock, like if an OR-split are synchronized by an AND-join. In addition, the recommender also validates if the model contains a lack of

Comparing	Feature	Measure	Weight
Places	name	synt sim.	0.2
	Attribute/Value	str sim.	0.5
	successor	ling sim.	0.3
Attributes	name	synt sim.	0.2
	sibling Attribute	ling sim.	0.3
	Values	ling sim.	0.3
	Place	ling sim.	0.2
Values	name	synt sim.	0.2
	Attribute	ling sim.	0.5
	Value reference	ling sim.	0.3
Transitions	name	synt sim.	0.2
	ToPlace	sling sim.	0.4
	FromPlace	ling sim.	0.4

Figure 3.20: The different proposed factors
Source: Betz et al. 2006

synchronization, if for example an AND-split is synchronized by an OR-join. This validation takes place by using reduction rules, where a structural error free model is one where the reduction result is an empty graph.

The authors also propose for further development of the recommender to consider a mechanism to learn from the user's previous behaviour to recommend a more personal oriented recommendations. The propose the use of a classification method to classify stored elements to partitions to which the user's behaviour can be compared to. The closest user behaviour in addition to the context matching and criteria mentioned above bring about a stronger user personal recommendation result. To capture the user's behaviour various techniques can be applied. Examples of them is Support Vector Machines SVM, Neural Nets, Bayesian Networks, Information Filtering and content-based Information Filtering.

3.1.18 Semantic based auto-completion of business process modelling in eGovernment

The recommender proposed in this paper Maalouf & Sokhn (2014) tackles the use of business process modeling in the public sector. The authors propose a business process modeling recommender tool to auto-complete BPMN business processes models being modeled by suggesting business process parts to the user as recommendations and the user chooses accordingly. Moreover, the recommender is cased for public sector service

in Switzerland. In details, Switzerland contains 26 cantons, multiple municipalities and four official languages German, French, Italian and Romansh. Given these multilayer dependencies, the authors had to propose an approach to make use of them to create an agile recommender applicable in all different cantons and municipalities.

The recommender, as mentioned previously, suggests to the users what comes next in the reference model. In details, this recommender is based on a semantic business process models database. The recommender queries this database to retrieve closely-matching business process models. Nevertheless, semantic business process management introduces the usage of ontological information and semantic web technologies to make business process modeling more context aware, thus simplifying the BPM processing complexity. Furthermore, the first phase of installing the recommender environment is utilizing a semantic business process models database. The authors use a database of business process models represented with Resource Description Framework (RDF). Resource Description Framework (RDF) is a semantic web ontology standard that can describe models using simple text notation. The use of RDF is perfect for fast querying as there exists a language for that called SPARQL. After having the database, they authors built an interface an open source modeling tool. The tool is ““Core Components”. This tool allows plugins for further enhancement and development of the recommender. Moreover, the reference model, in BPMN 2.0, being queried on this interface is transformed into RDF and then querying is utilized to retrieve the closest business processes are in the shared semantic business process models database. The queries can be in the forms of:

1. Label match querying, results include matches from different languages. This is solved with a translation mechanism that will be explained in the next paragraph.
2. Graphical match querying, for example an event E1 followed by Activity A1.
3. Role match querying: process owned/executed/published by person/role/entity.
4. Fragment match querying: look up all processes that contain a given fragment.
5. Connection match querying: look up all process that include two participants interaction.
6. Goal based match querying: lookup all process belonging to a certain goal.
7. Language based match querying: look up all languages in a given language.
8. The recommender uses SPARQL to query according to one of latter 7 query forms. Consequently, the SPARQL query syntax is used accordingly.

As mentioned before, Switzerland has four different languages and if it is required from this recommender to be generic and handle the whole of Switzerland. So, for the shared semantic business process model database, a translation mechanism is used to handle the languages difference. Due to German and French being the main two languages, the mechanism handles translation from German to French, thus having a standardized process repository. As mentioned previously, BPMN models are transformed into RDF.

The translation mechanism is applied before RDF transformation for the sake of language standardization. Furthermore, BPMN 2.0 standard enables XML format serialization of the models. These XMLs can contain the names and attributes of entities in business process models which in terms can be extended and used to acquire the corresponding entities names. In abstract, the user receives a translation suggestion for a certain entity and decides to accept it or not. If the translation is accepted then it's saved for reuse in similar labels. In details, translation process is brought about by using a local language data TERMDAT database formed in RDF. TERMDAT is a database for handling Swiss legal and administrative public sector technologies.Moreover, through TERMDAT the users' past language choices are saved. If the entity label is new and not seen before, then an implemented dictionary is accessed. This dictionary is GATE. GATE enables the possibility to search word in a text and matches it according to dictionary concepts. It only works on exact matches. Consequently, in the case of no matches are found then an external web translation API is utilized. In both cases, as mentioned before, the translation suggestion is shown to the user and if the user chooses it then its added to the TERMDAT local data database.

3.2 Comparison analysis

The last section described the 18 recommenders in details to present an overview of them. As for this sub-chapter, it enriches the survey of BPM recommender systems by introducing further comparison analysis on the recommender systems. It achieves that through implementing 4 comparison tables. The tables utilise the comparison analysis features mentioned in chapter 2. Furthermore, the tables are considered complementary to the information presented in the last sub-section. In other words, not all information mentioned here, are mentioned in the overview in the last sub-chapter.

Moreover, the comparison analysis tables are:

1. Method and Notation table- is used to compare the methods the recommenders are implemented with. In addition, it compares the BPM notations of the recommenders. It can be found in table 3.2.

2. Recommendation type table- describes the recommendation type features that each recommender possesses. It is found in table 3.3

3. Form, Repository and Common properties table- introduces the form of the recommenders and the type of models' repository they utilise. In addition, it sets the common properties for each recommender. Table 3.4 is used for this task.

4. Utilities and Distance measure table- sets the utilities and distance measures for the recommenders. This table's features are open text field. It is in table 3.5

Lastly, table 3.1 indexes the recommenders' publications to be used in the 4 comparison analysis tables.

	Publication
1	A Recommendation System to Facilitate Business Process Modeling
2	Assisting Business Process Design by Activity Neighborhood Context Matching
3	Context-Based Service Recommendation for Assisting Business Process Design
4	Action Patterns in Business Process Models
5	An Efficient Recommendation Method for Improving Business Process Modeling
6	Recommendation Based Process Modeling Support: Method and User Experience
7	Social Software for Modeling Business Processes
8	Advanced social feature in a recommendation system for process modeling
9	Application of Bayesian Networks to Recommendations in Business Process Modeling
10	Integration of Activity Modeller with Bayesian network based recommender for business processes
11	Context-Sensitive Textual Recommendations for Incomplete Process Model Elements
12	On the Automatic Labeling of Process Models
13	On the refactoring of activity labels in business process models
14	Supporting flexible processes through recommendations based on history
15	Business Process Models as a Showcase for Syntax-Based Assistance in Diagram Editors
16	Towards Auto-Suggested Process Modeling – Prototypical Development of an Auto-Suggest Component for Process Modeling Tools
17	Automatic user support for business process modeling
18	Semantic based auto-completion of business process modelling in eGovernment

Table 3.1: Publications index table

Publication	Concept														
	Method										Notation				
Feature	Association rule mining	Based on history	Bayesian networks	Deep learning/ Neural networks	Graph mining	Notation independent model	Semantic model	Support Vector Machine	Syntax-based assistance	Textual model	BPEL	BPMN	EPC	Petri-nets	XML
1					X									X	X
2					X							X			
3					X						X				
4	X												X		
5	X											X	X	X	
6								X						X	
7								X						X	
8								X						X	
9			X									X			
10			X									X			
11										X		X	X		
12										X			X		
13										X		X	X		
14		X													
15									X			X			
16						X					X	X	X	X	
17							X							X	
18							X					X			

Table 3.2: Method and Notation comparison table

Publication	Concept												
	Recommendation type												
	Attachment				Structural		Textual			Position-based			
Feature	Decision table	Link	Service task	Sub-processes	Single element	Structure of elements	Fullname suggestion	Name completion	Guard conditions	Forward	Auto-completion	Backward
1			X		X					X		
2		X	X		X					X		
3		X	X		X					X		
4		X	X		X					X		
5		X	X		X					X		X
6		X	X	X	X	X				X		
7		X	X	X	X	X				X		
8		X	X	X	X	X				X		
9		X	X		X					X	X	X
10		X	X		X					X	X	X
11							X	X	X			
12							X		X			
13							X		X			
14												
15		X	X	X	X	X				X	X	
16		X	X		X					X		
17		X	X		X					X		
18		X	X		X					X		

Table 3.3: Recommendation type comparison table

Publication	Concept														
	Form			Repository		Common properties									
Feature	Application	Extension/ plugin	Framework/ concept	Event logs	Models database	Social features	Personalization	Not domain specific	Accuracy/ confidence	Recall/ precision	Evaluation check	Has an example	Uses ontologies	Not fully auto-moated
1	X				X				X	X	X	X		
2			X		X						X	X		
3	X				X						X	X		
4			X		X		X	X	X		X	X		X
5			X		X						X			
6	X				X						X	X		
7	X				X	X					X	X		
8	X				X	X	X				X	X		
9			X		X				X			X		X
10		X			X				X		X	X		
11	X				X						X	X		
12			X									X		
13	X				X					X	X	X		
14	X			X							X	X		
15		X			X							X		
16	X				X									
17			X		X							X	X	
18	X				X							X	X	

Table 3.4: Form, Repository and Common properties comparison table

Publication	Concept	
	Utilities	Distance measure
1	-	MCSub, MCSub and xSED.
2	-	Levenshtein.
3	Java.	Levenshtein.
4	WordNet	-
5	gSpan	String Edit Distance on minDFS codes.
6	WordNet	Tf-idf.
7	WordNet	Tf-idf, Minkowski, Hamming distance, Pearson's correlation coefficient
8	WordNet	Tf-idf.
9	ME, SamIam	-
10	ME, SamIam, Activiti	Levenshtein.
11	Java, ANC corpus, Likert scale	Lin semantic. similarity
12	WordNet	-
13	WordNet, Stanford Parser	-
14	ProM, Java	-
15	DiaGen	-
16	JSON	Levenshtein
17	WordNet, OWL	Levenshtein
18	REF, SPARQL, TERMDAT, GATE	Semantic querying

Table 3.5: Utilities and Distance measure comparison table

4 BPM recommender systems survey results

The comparison analysis in the last chapter provides this master's thesis with an informative listed overview of the BPM recommender systems. This chapter, analyses each of the comparison tables. It achieves this by describing state of the recommenders in terms of the analysis features. Like for example, if a certain feature value is more frequent than the other or vice versa. In addition, this chapter provides suggestions for future work in the field of BPM recommender systems.

4.1 Comparison analysis findings

- Method: It is noticed from table 3.2 that the methods implemented for the recommendations are scattered and diverse. However, most of them are grouped by a certain method. For example:

  1. 3 recommenders are implemented with graph mining and 3 more with are implemented with textual models
  2. 2 recommenders utilize association rule mining, another 2 use Bayesian networks and 2 more are implemented with a semantic model.

  Moreover, it is noticed that deep learning and neural networks are not used for any recommender although they are considered a well-known recommendation approach.

- Notation: BPMN, Petri-nets, and EPC share the most frequent mentions in the recommenders' publications as mentioned in table 3.2. They are the most widely used graphical notations in BPM in general. As shown in the last chapter, most of the BPM recommender systems depend on the graphical feature in BP models to use it for recommendations. Therefore, BPMN, Petri-nets, and EPC are the most frequent notations in the recommenders.

- Recommendation type: Table 3.3 provides an overview of the recommendation types for the recommenders. Moreover, there are several points to tackle for recommender systems recommendations types:

  1. The majority of the attachment recommendation types are links and service tasks. In addition, most recommendations are single elements. This is due to the fact that most recommenders recommend a single element in the reference model. The element is either a link or a service task.
  2. It is noted, that most recommenders suggest the next coming element in the reference model. This leads to forward recommendations to be present in majority of the recommenders.

3. Publication 14, as indexed in table 3.1, does not cover any type of recommendation because it does not recommend new elements in the reference model. On the contrary, it suggests to the users the status of each existing model node from a certain goal that the user defines like efficiency maximization or cost reduction.

4. Only 3 recommenders provide textual recommendations.

- Form: As shown in table 3.4, more than half of the recommenders are applications. While 6 are concepts or frameworks proposed by their authors.

- Repository: As noted, in table 3.4, almost all the recommenders utilize a models database, while only 1 recommender uses event logs to generate recommendations.

- Common properties: As for the common properties, the following findings are observed from table 3.4:

  1. Most of the recommenders contain an example in their publication. While more than half of them performs an evaluation check to validate the recommenders they propose.

  2. Combining either social features or personalization with BPM recommender systems rarely occurs.

  3. Two recommenders mention that they lack full automation in the system they propose. It is also noticed, they are concepts and not a full application.

- Utilities: Table 3.5 shows a recurrent utility that is used in several recommenders (7 recommenders). This utility is WordNet. Furthermore, WordNet is a lexical database, therefore it is used in identifying the BP models' context which heavily depends on the textual labels of their elements.

- Distance measure: Levenshtein distance measure is a dominant approach for distance measure for BPM recommender systems. This, in addition to further distance measures, are shown in table 3.5.

4.2 Future work recommendations

The surveyed BPM recommender systems reviewed in this thesis share many innovative approaches in their task. Each group of them, has a unique way of identifying the recommendation method, the BPM context and the models similarity check approach. However, there are always room for improvement.

In this sub-chapter, some future work recommendations are proposed for future development in the field of BPM recommender systems. The following recommendations are based upon shortcomings noticed in the current recommenders:

1. The methods with which the BPM recommenders are built are few in numbers. Few alternatives of machine learning algorithms are explored for example like association rule mining and Bayesian networks. However, many other field in machine learning are not utilised in this field. Examples to that are neural networks, deep learning, k nearest neighbour and matrix factorization. These machine learning algorithms are used in other types of recommenders. Therefore, this thesis proposes the use of these methods in the implementation of BPM recommender systems.

2. As mentioned in the last sub-chapter, the usage of forward recommendations is dominant. However, the other types of recommendations (Auto-completion and Backward) are rarely covered in recommenders. Therefore, this master's thesis recommends for future work the implementation of recommenders that include all positions of recommendations. Thus, providing the user not only with what they will introduce to the models but an improvement to the elements already existing in the model. This is achieved by either backward or auto-completion.

3. The recommenders are segregated into either recommenders that suggest elements to the user or recommenders that suggests a textual change for BP model. Moreover, no recommender so far introduces both element and textual suggestions together. Although combining both produces a more integral recommender with more functionalities fitting the user's convenience and the quality of the BP models.

4. The field of BPM is a growing field and so is the recommendation technologies. So collaborating between the two will bring about strong enhancement to the field of BPM. Furthermore, this thesis reviewed 18 recommender systems where only 10 of them are an application. Therefore, it recommends more input in this field because of two fold. Firstly, the recommendation technologies in general are available and secondly the BPM recommender systems field is in need of more research.

5 Conclusion

In conclusion, BPM is a fresh field vastly used in enterprises and corporates. It handles organizational processes by representing the relations between them, usually graphically. Furthermore, BPM evolved over the years until it reached its current form. Nowadays, it aims at creating efficient BPM models to achieve effective results. The tremendous technological development in the last years introduced recommender systems in peoples' everyday life. Recommender systems are spread all over the online services, like for example Google's search auto-completion, Facebook's friends suggestion and YouTube's recommended videos.

Moreover, combining BPM with recommender systems ensures several enhancements in the modeling process and model quality. BPM recommendations provide more optimal, less error-prone and time and effort efficient models. Furthermore, the BPM recommendations come in different types. There are attachment recommendations which are attached to the existing BP model's elements. Another type is structural recommendations. They define whether a recommendation is a single element of a group of connected elements. Textual recommendations are also one of the different types a BPM recommendation can be. They represent the textual suggestions for the labels of the elements in the BP model. Lastly, the position-based recommendation describes whether a recommendation is put forward, backward or auto-completes the BP model.

The types of recommendations mentioned exist in current BPM recommender systems. However, it is not clear what the technologies are used, what methods are utilized and what the types of recommendations available for each recommender are. Therefore there is a need to survey the existing BPM recommender systems as it provides a detailed overview of the technologies existing and enriches the field to create enhanced BP models.

Furthermore, the methodology implemented in this master's thesis to perform a literature review follows a proposed framework. This literature review framework was a strong factor in retrieving high-quality BPM recommender systems publications. Moreover, it consists of 5 stages. In the first stage, the scope of the review is defined. The scope is defined by setting several variables. These variables are the Focus, Goal, Organisation, Perspective, Audience, and Coverage of the literature review. As for the second stage, the conceptualization of the topic is defined. During it, the research key terms are defined. For example, the key terms defined for this thesis were "BPM recommender systems", "Recommendation based bpm" and "Recommendation based bpm". Moreover, the third stage is the research phase. Around 240 publications were found, 18 of them were filtered to be used as the core of the BPM recommender systems survey. The publications were filtered mainly through reading their abstracts, analyzing their titles and searching for BPM recommender systems key terms in them. The fourth phase in the framework is the analysis and synthesis of the publications. During this phase, the approach for analyzing

64 the 18 recommender systems was set. The proposed approach to achieve this was by analyzing each recommender with certain features. If the recommender has that certain feature in it, then the feature will be set. The last stage is the research agenda, in which the reason for carrying out literature research is defined. Furthermore, the agenda for this literature research is to explore the existing BPM technologies for the enhancement of future development in the BPM recommender systems field.

Before the start of the survey, the analysis features over which each recommender is to be reviewed are set. They cover various areas in recommender systems. For example, the recommendation type of the recommender, the method it is implemented with, the BPM notation it covers and other various features. Furthermore, the survey takes place on two consecutive phases. In the first phase, a short detailed overview is given about each recommender. The method it is implemented with is described, and the types of recommendations it gives as well. In addition, the overview covers other areas as well like the BPM notations the recommender covers and in some the recommender evaluation process. After providing an overview for all the 18 recommenders, a comparison analysis takes place to complement the overview provided in the first phase. It achieves this by introducing 4 comparison analysis tables containing the analysis features mentioned before. Finally, the comparison analysis findings are reviewed and analyzed.

After the end of the survey process, this master's thesis reached some conclusions to be taken into consideration for future work in the field of BPM recommender systems. The survey showed that there are few methods with which the recommenders are implemented. In addition, the survey indicated that there is less work in the field as expected as the whole survey process is based on only 18 recommender systems publications. In this case, this thesis suggests using giving more input in this field by introducing new recommendation methods like neural networks, deep learning, k nearest neighbor and matrix factorization. The survey also a dominant direction with recommenders by recommending only the coming node in the model (forward recommendation). It also indicated that there are less textual recommendations than others. That's why this master's thesis strongly suggests building recommenders that provide recommendations in different positions in the BP model. In addition, this thesis recommends introducing textual recommendations more often to recommenders that only provide element recommendations. This creates a more integral recommender, which in turns strengthens the quality of the BP models

Bibliography

Betz, S., Klink, S., Koschmider, A. & Oberweis, A. (2006), Automatic user support for business process modeling, in 'IN: PROCEEDINGS OF THE WORKSHOP ON SEMANTICS FOR BUSINESS PROCESS MANAGEMENT', pp. 1–12.

Bobek, S., Baran, M., Kluza, K. & Nalepa, G. (2013), Application of bayesian networks to recommendations in business process modeling, Vol. 1101, pp. 41–50.

Bobek, S., Nalepa, G. J. & Grodzki, O. (2014), Integration of activity modeller with bayesian network based recommender for business processes, in 'KESE@ECAI'.

Chan, N. N., Gaaloul, W. & Tata, S. (2012), Assisting business process design by activity neighborhood context matching, in C. Liu, H. Ludwig, F. Toumani & Q. Yu, eds, 'Service-Oriented Computing', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 541–549.

Hornung, T. & Koschmider, Agnesand Lausen, G. (2008), Recommendation based process modeling support: Method and user experience, in Q. Li, S. Spaccapietra, E. Yu & A. Olivé, eds, 'Conceptual Modeling - ER 2008', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 265–278.

Kluza, K., Baran, M., Bobek, S. & Nalepa, G. J., eds (2013), Proc. Overview of Recommendation Techniques in Business Process Modeling, Koblenz, Germany.

Koschmider, A., Song, M. & A. Reijers, H. (2008), Social software for modeling business processes, Vol. 17, pp. 666–677.

Koschmider, A., Song, M. & Reijers, H. A. (2009), Advanced social features in a recommendation system for process modeling, in W. Abramowicz, ed., 'Business Information Systems', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 109–120.

Leopold, H., Mendling, J. & Reijers, H. A. (2011), On the automatic labeling of process models, in H. Mouratidis & C. Rolland, eds, 'Advanced Information Systems Engineering', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 512–520.

Leopold, H., Smirnov, S. & Mendling, J. (2012), 'On the refactoring of activity labels in business process models', Information Systems 37(5), 443 – 459. Best papers from DOLAP 2010.
URL: http://www.sciencedirect.com/science/article/pii/S0306437912000075

Li, Y., Cao, B., Xu, L., Yin, J., Deng, S., Yin, Y. & Wu, Z. (2014), 'An efficient recommendation method for improving business process modeling', IEEE Transactions on Industrial Informatics 10(1), 502–513.

Maalouf, E. & Sokhn, M. (2014), Semantic based auto-completion of business process modelling in egovernment, in E. Plödereder, L. Grunske, E. Schneider & D. Ull, eds, 'Informatik 2014', Gesellschaft für Informatik e.V., Bonn, pp. 1107–1118.

Mazanek, S. & Minas, M. (2009), Business process models as a showcase for syntax-based assistance in diagram editors, in A. Schürr & B. Selic, eds, 'Model Driven Engineering Languages and Systems', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 322–336.

Nguyen, C., Gaaloul, W. & Tata, S. (2011), Context-based service recommendation for assisting business process design, Vol. 85, pp. 39–51.

Pittke, F., Richetti, P. H. P., Mendling, J. & Baião, F. A. (2015), Context-sensitive textual recommendations for incomplete process model elements, in H. R. Motahari-Nezhad, J. Recker & M. Weidlich, eds, 'Business Process Management', Springer International Publishing, Cham, pp. 189–197.

Schonenberg, H., Weber, B., van Dongen, B. & van der Aalst, W. (2008), Supporting flexible processes through recommendations based on history, in M. Dumas, M. Reichert & M.-C. Shan, eds, 'Business Process Management', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 51–66.

Shitkova, M., Clever, N., Holler, J. & Becker, J. (2013), Towards auto-suggested process modeling – prototypical development of an auto-suggest component for process modeling tools.

Shuiguang Deng, Dongjing Wang, Y. L. B. C. J. Y. Z. W. M. Z. (2017), 'A recommendation system to facilitate business process modeling', IEEE Transactions on Cybernetics 47, 1380 – 1394.
URL: https://ieeexplore.ieee.org/document/7447788

Smirnov, S., Weidlich, M., Mendling, J. & Weske, M. (2009), Action patterns in business process models, in L. Baresi, C.-H. Chi & J. Suzuki, eds, 'Service-Oriented Computing', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 115–129.

vom Brocke, J. & Rosemann, M. (2014), Wiley Encyclopedia of Management, chapter Business Process Management.

vom Brocke, J., Simons, A., Niehaves, B., Riemer, K., Plattfaut, R. & Cleven, A., eds (2009), Proc. Reconstructing the Giant: On the Importance of Rigour in Documenting the Literature Search Process, Verona, Italy.