MANAGING BUSINESS PROCESS FLEXIBILITY AND REUSE THROUGH BUSINESS PROCESS LINES

Nicola Boffoli, Marta Cimitile and Fabrizio Maria Maggi
Computer Science Department, University of Bari,Via Orabona 4, Bari, Italy

Keywords: Business Process Lines, Process Flexibility, Process Reuse.

Abstract: The ever greater pressure of competition to which enterprises are subjected has made the process continuous improvement a crucial issue today. For this reason it should be useful to compose the business processes reusing previously modeled business process parts characterizing them according to the current market needs. This work presents an approach based on the use of Business Process Lines (BPL) to compose and characterize a business process according to different contexts reusing existing process parts. The approach has been applied to realize a BPL for the Software extraordinary maintenance. This BPL can be used to model different process variants of the Software extraordinary maintenance processes corresponding to different context profiles. The results demonstrate the approach applicability in a real case and underline that it allows to reuse and specialize the same process parts for many different contexts.

1 INTRODUCTION

Working in a highly competitive and ever changing environment, enterprises have to improve their processes continuously. It often happens that the existing business processes become inadequate due to the natural changeability and competitiveness of the business world. For this reason it's necessary to modify the business processes to tailor them according to different context factors (objectives, technologies, industrial standards, quality programs, budget, workers, tools, cultural factors) (Lindvall, 2000). Unfortunately, changing well defined and predictable processes often results very complex. In particular it can imply high costs and risks, above all in critical environments (small companies, strict schedule, poor resources). That's why it's necessary to define new approaches aiming to reduce the costs and mitigate the risks in process modeling, updating and continuous improvement. This work addresses this problem proposing an approach to make business processes reusable and flexible whenever the context changes. Our idea is that the reusability can be obtained individuating a number of process parts to be reused in different operative contexts. On the other hand, we suppose that the flexibility could be improved composing and characterizing these process parts to obtain different variants of the same process for different contexts.

For this purpose it is useful :

- Characterize all the possible operative contexts where the process could be executed;
- Identify all the process variants of the business process to be used in the different operative contexts;
- Identify the common parts of the different process variants;
- Identify their variable parts;
- Determine the composition rules driving the process engineer to choice the process parts suitable for a given context and integrate them in the corresponding process variant.

We can realize all this using the concept of BPL defined in (Boffoli, 2008) that transfers to the business processes the peculiarities typical of the Software Product Lines paradigm (Clements, 2001). A BPL is a set of similar business processes sharing a common part and characterized by different variant parts. Each process of a BPL can be applied in a specific context and is modeled composing activities, inputs and outputs needed in that specific context. Besides a conceptual definition of BPL, the proposed approach is also composed by a logical and an operative model. These models drive the process engineer to quickly identify and characterize a business process according to his own needs. In this way, the approach allows to reuse process parts

Boffoli N., Cimitile M. and Maggi F. (2009).
MANAGING BUSINESS PROCESS FLEXIBILITY AND REUSE THROUGH BUSINESS PROCESS LINES.
In *Proceedings of the 4th International Conference on Software and Data Technologies*, pages 61-68
DOI: 10.5220/0002257800610068
Copyright © SciTePress

and to make flexible the processes adapting them quickly to the environment changes.

The remainder of this paper is structured as follows: section 2 recalls the related works; section 3 describes the BPL approach; in section 4 the BPL approach is applied in a real case; section 5 completes the paper providing some conclusive insights and final remarks and showing prospective works.

2 RELATED WORKS

In the last years many authors have addressed the issues of process flexibility and reuse (Adams, 2006), (Kim, 2007) proposing approaches to support dynamic evolution in the business processes. In (Reichert, 2005), (Pesic, 2007) the authors propose methods to realize a workflow management system supporting flexible process changing. In particular ADEPT (Reichert, 2005) provides powerful mechanisms that allows to change process models (by inserting, moving or deleting activities) during execution. On the other hand DECLARE (Pesic, 2007) uses constraints based process model languages for the development of declarative models describing loosely structured processes. Nevertheless these works regard the process flexibility at run-time and propose approaches to manage different process instances and different possible control flows inside an only one process. Our approach on the contrary aims to select different process models according to the different context characteristics. Of course the above mentioned works can be used for the workflow management systems specification. Nevertheless, dealing with processes at execution level, they can't be used by domain experts to model business processes and define process requirements. Our idea is on the contrary to support managers (that in general are not technicians) through a repertoire of process parts at a higher abstraction level to compose a business process model for the process requirements specification.

In literature there are also approaches addressing the issue of process flexibility suggesting methods based on the adoption of Knowledge Bases (XU Ru-Zhi, 2005), (Malone, 2003). In particular in (XU Ru-Zhi, 2005) the authors propose a reuse-oriented process components framework for the reuse and retrieval of process components. It uses a facet-based process component classification scheme and XML based process description for the selection of the process component contained in a repository. Even if this approach allows the component reuse, it doesn't offer a support to the process characterization according to the context peculiarities. Our approach on the contrary allows to model, through the BPL, the operative context and the relationships between different context profiles and the specific elements to be inserted in the process model. In this way, it allows to analyze better the contexts, the processes and the relationships among them, and to store and transfer the knowledge contained in these relationships. Finally MIT Library (Malone, 2003) is a large collection of business processes about different application domains. Here is just mentioned a context modeling but the approach isn't operative at all.

3 THE BPL APPROACH

To describe the BPL approach we introduce a conceptual model, a logical model and an operative model. The conceptual model explains the BPL definition and its conceptual meaning. In the logical model we introduce functions useful to select, specialize and integrate existing process parts to obtain the process variant more suitable for a specified context. Basing on the logical model, the operative model is implemented by using decision tables that support the process engineer to identify the suitable process variant.

3.1 BPL Conceptual Model

A BPL is a set of similar business processes sharing a common part (*commonality*) and characterized by a variant part (*variability*) depending on the specific context where the process will be applied. So, a BPL works integrating a set of *process assets*, i.e. atomic reusable parts of a business process (one or more activities with their IN/OUT). In particular the commonality is a set of invariant assets and the variability is a set of variant assets selected according to a fixed context profile. Commonality and variability are then integrated in order to obtain a process variant to be applied in the fixed context. The assets integration rules are driven by their IN/OUT artefacts allowing to establish the succession of the process assets: the outputs of the previous asset are the inputs of the successive one.

When a BPL is selected the invariant assets and all the candidate variant assets are specified. The BPL is selected on the basis of the process that has to be modeled: for example if we are interested to

the process "Sell" we will use a "Sell BPL". The invariant assets of the "Sell BPL" can be for example "Obtain Order", "Deliver product" and "Receive Payment". Each one of these process assets is composed by the basic activities, inputs and outputs needed for its own purpose within the "Sell" process. Afterwards, to identify a specific process variant the variant assets has to be selected among the candidate variant assets on the basis of a specific context. For example if we want to sell via electronic store the asset "Organize the web site" has to be selected. The process assets (variant or invariant) have to be then specialized on the basis of the context itself. The specialization aims to specify the behaviour of each process asset applying on it some specializing actions. In particular, an asset can be specialized adding IN/OUT artefacts to an activity, specializing an artefact, specializing an activity, adding an attribute to an artefact (size, compilation guideline, quality standards etc.), adding an attribute to an activity (required skills, tools, hardware or software resources etc.). For example if we want to sell via electronic store the activity "Receive Payment" can be specialized in "Receive online Payment". Finally the process assets selected and specialized are integrated in the required process variant. The conceptual model activities are synthesized in Figure 1.

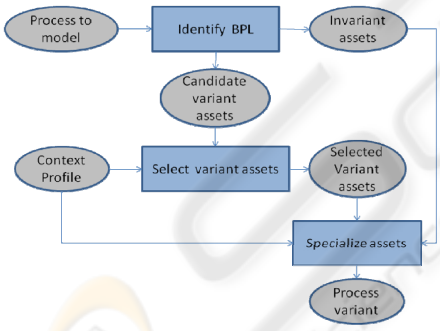


Figure 1: Conceptual Model.

3.2 BPL Logical Model

To chose the suitable process variant of a BPL to be applied in a specific context profile, we need a function f associating to a specific context profile the process variant specific for the given context:

$$f: CP \rightarrow S \qquad (1)$$

where

- CP is the set of all the possible context profiles. An element $cp \in CP$ is represented as a vector of instantiated diversity factors DF_i i=1, …, r. Each DF_i is a factor characterizing a particular aspect of the environment and has a definition domain $[DF_i]=\{df_{i1}, df_{i2}, \ldots, df_{iq}\}$ where each df_{ij} j=1,...q is an instance of DF_i. So we can say that the set CP is: $CP= [DF_1] \times [DF_2] \times \ldots\ldots \times [DF_r]$.
- S is the set of all the possible process variants of the considered BPL.

f can be detailed in this way:

$$f(cp)= \Phi(\sigma(cp, K), \sigma(cp, \chi(cp))) \qquad (2)$$

If A is the set of all the process assets associated to the BPL, in (2):

- $K = \{ia_1,ia_2\ldots..ia_n\} \subseteq A$ is the set of invariant assets realizing the commonality;
- $\chi: CP \rightarrow CVA=A-K$ is the function associating to each fixed $cp \in CP$ the set of variant assets $\{va_1,va_2\ldots..va_m\} \subseteq CVA$ realizing the process variability according to the fixed context cp. CVA=A-K is the set of the candidate variant assets associated to the BPL;
- σ is the function including the assets specialization rules on the basis of the context profile cp. It associates to a set of assets another set of assets specialized according to the fixed context profile. In particular $\sigma(cp, \{asset_1,asset_2\ldots..asset_p\}) = \{\sigma_1(cp,asset_1), \sigma_2(cp,asset_2),\ldots, \sigma_n(cp,asset_p)\}$, where $\sigma_1, \sigma_2,\ldots, \sigma_p$ are transformations specializing respectively the assets $asset_1$, $asset_2$,..., $asset_p$ according to the context profile cp.
- Φ includes the integration rules useful to compose the commonality to the variability specialized according to the context profile cp.

So, $f(cp) = \Phi(\sigma(cp, K), \sigma(cp, \chi(cp))) = \Phi(\sigma(cp, \{ia_1,ia_2\ldots,ia_n\}), \sigma(cp, \{va_1,va_2..,va_m\})) = \Phi(\{\sigma_1(cp, ia_1), \sigma_2(cp, ia_2), \ldots, \sigma_n(cp,ia_n)\}, \{\sigma_{n+1}(cp, va_1), \ldots, \sigma_{n+m}(cp,va_m)\})$ identifies the process variant suitable for the context profile cp.

3.3 BPL Operative Model

A process variant of a BPL is obtained performing two main activities:

- *Variability Selection* to select, according to a specific context profile, the suitable variant assets.
- *Asset Specialization* to specialize each process asset basing on the fixed context profile.

These activities can be driven through a decision tables system.

A decision table is a tabular representation of the decision-making task, where the state of a set of conditions implies the execution of a set of actions (Maes, 1988), (Fayyad, 1996), (Vanthienen, 1998). In general a decision table has four quadrants: conditions (Cond), conditional states (S), actions (Act) and rules (x). The table is defined so that each combination of conditions and conditional states corresponds to a set of actions to execute.

According to the variability selection and asset specialization activities, two different kinds of decision tables are proposed: the *variability selection tables* and the *asset specialization tables*.

A *variability selection table* implements the function χ of the logical model. It allows to select the suitable variant assets composing the variability characteristic of a specified context profile. So this kind of decision table is structured as follows (Figure 2):

- the CONDITION quadrant contains the diversity factors DF_i i=1,...r driving the variant assets selection;
- the CONDITIONAL STATE quadrant contains the possible value of each diversity factor: $[DF_i]=\{df_{i1}, df_{i2}, \ldots, df_{iq}\}$;
- the ACTION quadrant contains all the candidate variant assets ($\in$CVA) that can be selected to realize the process variability;
- the RULE quadrant identifies the relationships between each context profile and the variant assets to realize the corresponding process variability.

DF_1	df_{11}				df_{12}			
DF_2	df_{21}		df_{22}		df_{21}		df_{22}	
...	.....							
DF_r	df_{r1}	df_{r2}	df_{r1}	df_{r2}	df_{r1}	df_{r2}	df_{r1}	df_{r2}
va_1	-	-	X	-	-	-	X	-
va_2	X	-	X	-	X	-	X	-
va_3	-	-	-	-	X	-	X	-
va_4	X	-	X	-	X	-	X	-
va_5	-	-	X	X	-	X	-	X
.....	.....							
va_m	X	-	X	-	-	X	-	X

Figure 2: Variability Selection table.

An *asset specialization table* implements a function σ_i of the logical model. It allows to specialize a process asset (variant or invariant) on the basis of specified context profile, executing a set of specializing actions. So this kind of decision table is structured as follows (Figure 3):

- the CONDITION quadrant contains the diversity factors DF_i i=1,...r driving the asset specialization;
- the CONDITIONAL STATE quadrant contains the possible values of each diversity factor: $[DF_i]=\{df_{i1}, df_{i2}, \ldots, df_{iq}\}$;
- the ACTION quadrant contains the actions to specialize the asset according to the specified context profile;
- the RULE quadrant identifies the relationships between each context profile and the specializing actions to be applied.

DF_1	df_{11}				df_{12}			
DF_2	df_{21}		df_{22}		df_{21}		df_{22}	
...	.....							
DF_n	df_{n1}	df_{n2}	df_{n1}	df_{n2}	df_{n1}	df_{n2}	df_{n1}	df_{n2}
sa_1	-	x	x	-	x	-	x	x
sa_2	x	-	x	-	-	x	-	x
.....	.....							
sa_r	-	x	-	x	-	x	-	x

Figure 3: Asset Specialization table.

Finally the function Φ of the logical model allows to compose the process assets selected and specialized through the above described decision tables. The assets integration rules are driven by their IN/OUT artefacts as explained before.

4 CASE STUDY: A BPL FOR SOFTWARE EXTRAORDINARY MAINTENANCE

A Software extraordinary maintenance process is a sequence of activities aiming to identify and remove obsolete components of a software application, to increase the components maintainability and to extract knowledge for their improvement.

Over the years Software extraordinary maintenance processes have been characterized by continuous changes followed by the variable size and complexity of software products. For this reason the Software extraordinary maintenance processes represent a field of interest for the BPL application.

So, according to the proposed approach, we will formalize the Software extraordinary maintenance processes using a BPL. The proposed BPL is obtained by the literature analysis but it can be improved and changed with use.

4.1 BPL Definition

Starting from the analysis of Software the maintenance processes (Rugaber, 1994), (Rugaber, 1995), (Rugaber, 2006), we have extracted their common parts. In this way we have identified the

invariant assets realizing the BPL commonality (Table 1).

Table 1: Invariant assets.

ia_1	*Inventory*
ia_2	*Data Classification*
ia_3	*Data Reconstruction*

Afterwards to define the variability selection table we have modeled the context defining a number of diversity factors and for each factor all the possible values. The identified diversity factors are shown in Table 2.

Table 2: Software extraordinary maintenance diversity factors.

Diversity Factors	**Values**
Documentation State	*Updated, Obsolete*
Code Readability	*Good, Not Good*
Variable Naming	*Good, Not Good*
Maintenance Cost	*Low, High*
Code Complexity	*#McCabe>7,#McCabe<=7*
Dead Data	*Present, Not Present*
Dead Instructions	*Present, Not Present*
# KNOTS	*>3, <=3*
Information Hiding	*Yes, No*
Data Banker	*Yes, No*
System Performance Improvement	*Required, Not Required*
In vivo	*Y, N*
Kind of Database Redesign	*Structure, Type, Structure and Type*

Finally we have identified the candidate variant assets that can be selected for each possible context profile (Table 3).

Table 3: Candidate variant assets.

va_1	*Database Redesign*
va_2	*Legacy Restoring*
va_3	*Data Migration*
va_4	*Restoring Equivalence Test*
va_5	*Requirements Reconstruction*
va_6	*Structure Chart Reconstruction*
va_7	*Procedures Reengineering*
va_8	*Residual Database Emptying*
va_9	*Metadata Cleaning*
va_{10}	*Reengineering Equivalence Testing*
va_{11}	*Documentation Reconstruction*

Each process asset is represented as one or more activities and their IN/OUT. For example the asset va_3 ="Data Migration" is represented as in Figure 4.

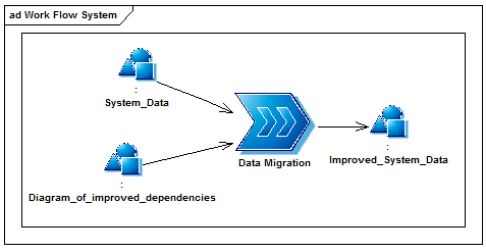


Figure 4: "Data Migration" asset.

In this way we can realize the variability selection table (Figure 5). It encloses the rules to associate to each possible context profile the related variant assets. These ones have to be composed with the BPL invariant assets to obtain the process variant specific for the specified context profile.

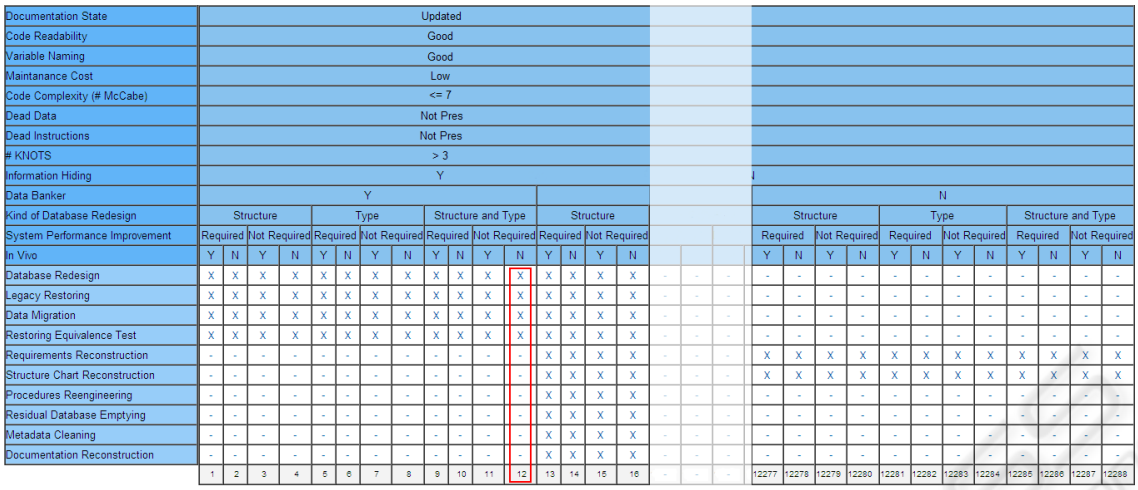
For example, considering the column 12 of the variability selection table, we have that for the context profile cp*= (Updated, Good, Good, Low, #McCabe<=7, Not Present, Not Present, #KNOTS>3, Yes, Yes, Structure and Type, Not Required, No) the variant assets to select are: va_1= "Database Redesign", va_2= "Legacy Restoring", va_3= "Data Migration", va_4= "Restoring Equivalent Test".

So to obtain the process variant specific for the given context profile we have to compose these assets with the invariant assets of the commonality (ia_1= "Inventory", ia_2= "Data Classification" and ia_3= "Data Reconstruction").

Moreover, to define the asset specialization tables, for ech process asset we have identified the specializing actions corresponding to each possible context profile. For example for the asset "Data Migration" the specializing actions shown in Table 4 are identified.

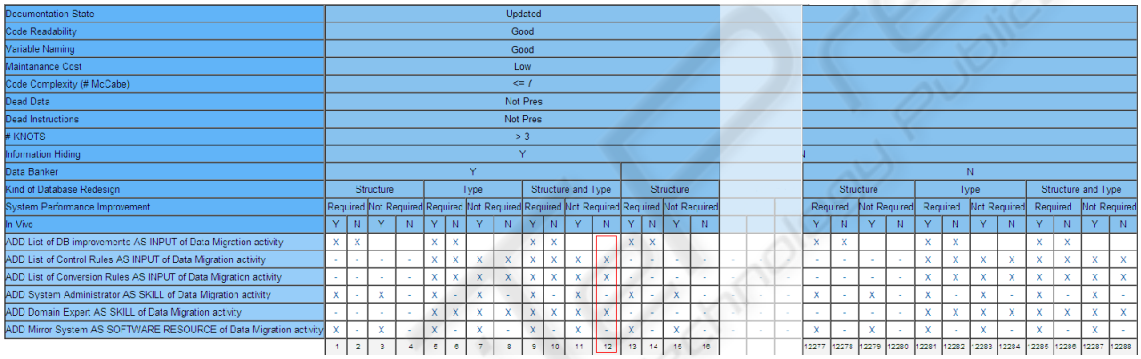
Table 4: Specializing action for the asset "Data Migration".

sa_1	*ADD List of DB improvements AS INPUT of Data Migration activity*
sa_2	*ADD List of Control Rules AS INPUT of Data Migration activity*
sa_3	*ADD List of Conversion Rules AS INPUTof Data Migration activity*
sa_4	*ADD System Administrator AS SKILL of Data Migration activity*
sa_5	*ADD Domain Expert AS SKILL of Data Migration activity*
sa_6	*ADD Mirror System AS SOFTWARE RESOURCE of Data Migration activity*



Documentation State	Updated																													
Code Readability	Good																													
Variable Naming	Good																													
Maintanance Cost	Low																													
Code Complexity (# McCabe)	<= 7																													
Dead Data	Not Pres																													
Dead Instructions	Not Pres																													
# KNOTS	> 3																													
Information Hiding	Y																													
Data Banker	Y																				N									
Kind of Database Redesign	Structure				Type				Structure and Type				Structure								Structure				Type				Structure and Type			
System Performance Improvement	Required		Not Required		Required		Not Required		Required		Not Required		Required		Not Required						Required		Not Required		Required		Not Required		Required		Not Required	
In Vivo	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N					Y	N	Y	N	Y	N	Y	N	Y	N	Y	N
Database Redesign	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	-	-	-		-	-	-	-	-	-	-	-	-	-	-	-
Legacy Restoring	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	-	-	-		-	-	-	-	-	-	-	-	-	-	-	-
Data Migration	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	-	-	-		-	-	-	-	-	-	-	-	-	-	-	-
Restoring Equivalence Test	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	-	-	-		-	-	-	-	-	-	-	-	-	-	-	-
Requirements Reconstruction	-	-	-	-	-	-	-	-	-	-	-	-	X	X	X	X	-	-	-		X	X	X	X	X	X	X	X	X	X	X	X
Structure Chart Reconstruction	-	-	-	-	-	-	-	-	-	-	-	-	X	X	X	X	-	-	-		X	X	X	X	X	X	X	X	X	X	X	X
Procedures Reengineering	-	-	-	-	-	-	-	-	-	-	-	-	X	X	X	X	-	-	-		-	-	-	-	-	-	-	-	-	-	-	-
Residual Database Emptying	-	-	-	-	-	-	-	-	-	-	-	-	X	X	X	X	-	-	-		-	-	-	-	-	-	-	-	-	-	-	-
Metadata Cleaning	-	-	-	-	-	-	-	-	-	-	-	-	X	X	X	X	-	-	-		-	-	-	-	-	-	-	-	-	-	-	-
Documentation Reconstruction	-	-	-	-	-	-	-	-	-	-	-	-	X	X	X	X	-	-	-		-	-	-	-	-	-	-	-	-	-	-	-
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	-	-	-		12277	12278	12279	12280	12281	12282	12283	12284	12285	12286	12287	12288

Figure 5: Sample of Variability Selection table.



Documentation State	Updated																													
Code Readability	Good																													
Variable Naming	Good																													
Maintanance Cost	Low																													
Code Complexity (# McCabe)	<= 7																													
Dead Data	Not Pres																													
Dead Instructions	Not Pres																													
# KNOTS	> 3																													
Information Hiding	Y																													
Data Banker	Y																				N									
Kind of Database Redesign	Structure				Type				Structure and Type				Structure								Structure				Type				Structure and Type			
System Performance Improvement	Required		Not Required		Required		Not Required		Required		Not Required		Required		Not Required						Required		Not Required		Required		Not Required		Required		Not Required	
In Vivo	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N					Y	N	Y	N	Y	N	Y	N	Y	N	Y	N
ADD List of DB improvements AS INPUT of Data Migration activity	X	X			X	X			X	X			X	X							X	X			X	X			X	X		
ADD List of Control Rules AS INPUT of Data Migration activity	-	-	-	-	X	X	X	X	X	X	X	X	-	-	-	-	-	-	-		-	-	-	-	X	X	X	X	X	X	X	X
ADD List of Conversion Rules AS INPUT of Data Migration activity	-	-	-	-	X	X	X	X	X	X	X	X	-	-	-	-	-	-	-		-	-	-	-	X	X	X	X	X	X	X	X
ADD System Administrator AS SKILL of Data Migration activity	X	-	X	-	X	-	X	-	X	-	X	-	X	-	X	-	-	-	-		X	-	X	-	X	-	X	-	X	-	X	-
ADD Domain Expert AS SKILL of Data Migration activity	-	-	-	-	X	X	X	X	X	X	X	X	-	-	-	-	-	-	-		-	-	-	-	X	X	X	X	X	X	X	X
ADD Mirror System AS SOFTWARE RESOURCE of Data Migration activity	X	-	X	-	X	-	X	-	X	-	X	-	X	-	X	-	-	-	-		X	-	X	-	X	-	X	-	X	-	X	-
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	-	-	-		12277	12278	12279	12280	12281	12282	12283	12284	12285	12286	12287	12288

Figure 6: Sample of Asset Specialization table.

The asset specialization table (Figure 6) encloses the rules to associate to each context profile, the actions to be executed to specialize the behaviour of the considered asset.

According to the context profile cp* considered before (column 12), we need to add List of Control Rules and List of Conversion Rules as inputs of Data Migration activity and to require a Domain Expert as skill for the activity execution (Figure 7).

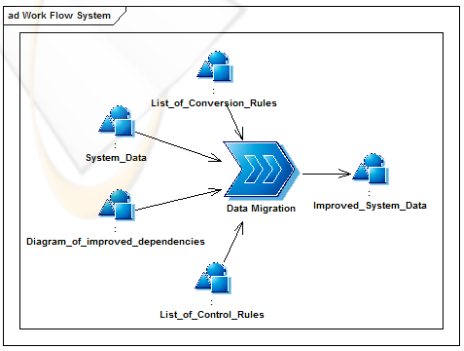


Figure 7: Specialized "Data Migration" asset.

4.2 BPL Application

The obtained BPL can be used to model some variants of the Software extraordinary maintenance process.

According to the context profile cp* considered before (column 12) we have obtained the process variant represented in Figure 8. The process includes the BPL commonality and the variant assets identified on the basis of the context profile cp* using the variability selection table. Each asset is specialized with the related asset specialization tables before being integrated in the process variant.

4.3 Discussion

In table 5 some synthesis data for the considered case study are reported. In the defined BPL the commonality is composed by 3 invariant assets and there are 11 candidate variant assets. For each asset there are on average 7,3 specializing actions. The BPL manages 12288 different contexts. Each asset is

reused on average 6353,4 times in the different contexts. On the one hand, these data show that each variant asset is reused many times. It gives a measure of the capacity of the proposed approach to reuse the same assets in different contexts. On the other hand, the flexibility is assured by the large number of specialization actions related to each considered assets. So starting from 14 assets and their specializations, the proposed approach allows to consider 12288 different contexts.

Moreover the BPL approach gives other effective advantages:

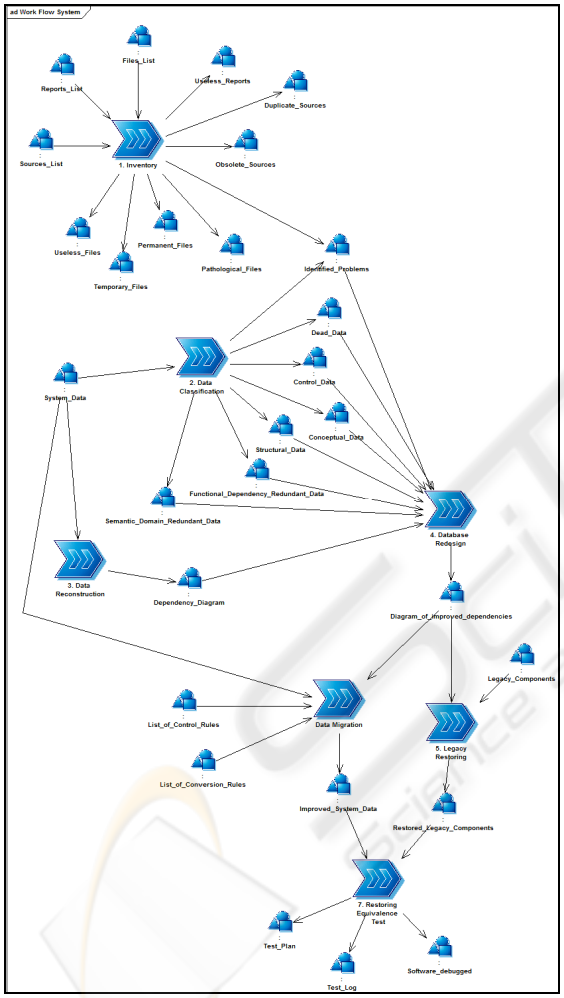


Figure 8: Example of process variant.

- a BPL can be easily maintained and continuously improved. The decision tables system can easily evolve when new experiences are acquired. For example in the proposed case study it is possible, on the basis of user feedbacks add new conditions or new actions to the decision tables;
- a BPL reduces the engineering effort to model and modify a business process: modeling a little number of assets, it allows to obtain many process variants suitable to different specific contexts;
- a BPL increases the total number of business processes that can be effectively modeled and managed allowing to consider all the possible context profiles;
- a BPL increases process quality and reduces risk in process execution. In fact the consistency of the business processes extracted from a BPL is assured by the consistency of the decision tables system implementing it. Moreover the reusable process parts used to obtain the process variants are already validated in other similar contexts.

Table 5: Synthesis data.

Specialization Actions for each asset	Variant Assets	Invariant Assets	Context Variants	#Reuses
7,3	11	3	12288	6353,4

5 CONCLUSIONS

The work proposes an approach to manage business process reuse and flexibility based on the use of BPL. The approach has been applied in a real case defining a Software extraordinary maintenance BPL.

The BPL is obtained by a Software extraordinary maintenance literature analysis. It supports and facilitates the reuse and the specialization of a set of identified assets in different contexts. In particular the composition of 14 assets allows to identify different process variants suitable for 12288 different contexts. Moreover the conditions and consequently the context profiles could be increased or improved with use. The BPL gradual increasing and changing allow to adequate the BPL to new market needs. These results prove that the BPL approach reduces the effort to model, update and continuously improve the business processes, making them flexible to the context changes. Moreover reusing processes parts already applied and improving continuously the decision tables reduce the risk of errors and increase the processes quality.

As future works, we will conduct empirical investigations to generalize the obtained results, to improve the proposed model and to confirm it

empirically. In particular we would use the business processes placed at disposal by (Malone, 2003) to define and apply a certain number of BPL. That's why we are interested to share our research experience with other researchers.

REFERENCES

Adams, M., ter Hofstede, A. H. M., Edmond, D., van der Aalst, W. M. P., 2006. Implementing Dynamic Flexibility in Workflow using Worklets. *BPM Center Report BPM-06-06.* BPMcenter.

Boffoli, N., Caivano, D., Castelluccia, D. Maggi, F. M., Visaggio, G., 2008. Business Process Lines for SOA Development through the Software Product Lines Paradigm. In *Proceedings of 12th Conference on Software Product Line (SPLC).* IEEE Computer Society.

Clements, P., Northrop, L., 2001. *Software Product Lines: Practices and Pattern,* SEI Series in Software Engineering. Addison–Wesley.

Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R., 1996. *Advances in Knowledge Discovery and Data Mining,* AAAI Press.

Kim, D., Kim, M., Kim, H., 2007. Dynamic Business Process Management based on Process Change Patterns. In *Proceedings International Conference on Convergence Information Technology (ICCIT 2007),* IEEE Computer Society.

Lindvall, M., Rus, I., 2000. Process Diversity in software Development. IEEE Computer Society.

Maes, R., Van Dijk, J.E.M., 1988. On the Role of Ambiguity and Incompleteness in the Design of Decision Tables and Rule-Based Systems. The Computer Journal.

Malone, T. W., Crowston, K., Herman, G. A., 2003. *Organizing Business Knowledge-The MIT Process Handbook,* MIT Press Cambridge.

Pesic, M., Schonenberg, H., van der Aalst W. M. P., 2007. DECLARE: Full Support for Loosely-Structured Processes. In *Proceedings of 11th IEEE International In Enterprise Distributed Object Computing Conference (EDOC 2007).* IEEE Computer Society.

Reichert, M., Rinderle, S., Dadam, P., 2005. Adaptive Process Management with ADEPT2. In *Proceedings of the 21st International Conference on Data Engineering (ICDE).* IEEE Computer Society.

Rugaber, S., DeBaud, J.M., Moopen, B., 1994. Domain Analisys and Reverse Engineering. In *Proceedings of the International Conference on Software Maintenance (ICSM94).* IEEE Computer Society.

Rugaber, S., Stirewalt, K., Wills, L. M., 1995. Direct Interleaving. In *Proceedings of the Conference on Software Maintenance (ICSM95).* IEEE Computer Society.

Rugaber, S., Harel, N., Govindharaj, S., Jerding, D. 2006. Problems Modeling Web Sites and User Behavior. In *Proceedings of the Eighth IEEE international Symposium on Web Site Evolution.* IEEE Computer Society.

XU Ru-Zhi, He Tao, CHU Dong Sheng, XUE Yun-Jiao, QIAN Le-Qiu, 2005. Reuse-Oriented Process Component Representation Retrieval. In *Proceedings of the 5st International Conference on Computer and Information Technology (CIT05*). IEEE Computer Society.

Vanthienen, J., Mues, C., Wets, G., Delaere, K., 1998. A tool-supported approach to inter-tabular verification. Expert Systems with Applications.