A Conceptual Modeling Framework for Software-Enabled Enterprise Transformation

Zia Babar

Faculty of Information, University of Toronto
zia.babar@mail.utoronto.ca

Abstract. Enterprises today increasingly rely on software to enable their operations. Leveraging recent advances in software and information technologies, many organizations are undertaking transformations that significantly alter their processes, artifacts, and social organizational relationships simultaneously and at multiple levels. Conventional conceptual modeling techniques are limited in their ability to visualize and reason about the complexities involved in the design of multi-faceted sociotechnical software systems and business processes while considering enterprise requirements for ongoing change and transformation. This PhD research project aims to develop a conceptual modeling framework to support the analysis and design of such transformations. A framework is proposed that consists of a set of modeling constructs, notations and methods that collectively allow for the exploration, analysis, and evaluation of alternative designs along multiple transformation dimensions.

Keywords: Conceptual Modeling, Enterprise Modeling, Goal Modeling, Requirements Engineering, Business Process Management.

1 Introduction

Enterprises undergo many transformations as they continually innovate to make better use of software and emerging digital technologies [1]. These transformations relate to business and software processes, software artifacts and tools, and organizational relationships. Through the increased use of software, enterprises are able to have more rapid cycles of tools and artifacts development and deployment, which help attain diverse enterprise objectives. Such transformation exercises need to be considered in a structured manner, particularly as enterprises are complicated entities with multifaceted social, process and technological elements with numerous uncertainties that make decision making difficult.

Contemporary conceptual modeling techniques that study and analyze enterprise architecture and business processes are generally used for static as-is and to-be representations of the enterprise. They assume fairly settled and stable set of requirements with limited support for catering to periodic, variable and continuous change, including the ability to decide between multiple alternate enterprise process and software systems configurations at run-time. Further, process design activities cannot be done once and

be assumed to be valid over large periods of time; the design would need to be periodically reviewed and reconsidered. Specific drivers of change (at a process, technological, and social level) need to be continuously monitored and evaluated, with appropriate selection of alternatives for ongoing implementation and execution. In transforming the enterprise to take advantage of digital technologies, there will be changes within the processes, as well as, changes to the relationships between processes.

2 Research Objectives

While such a focus on enterprise transformation can be studied from multiple perspectives, this PhD research project attempts to determine the design implications of an architecture of business processes with regards to enterprise goals for transformation and change. Through the identification of common phenomena (present in enterprises undergoing ongoing software-enabled change and transformation), a set of requirements are determined that would influence the development of a conceptual modeling framework for representing this process architecture. Such a framework reasons about several process architecture design possibilities for a given domain, possibilities that still allow the enterprise to attain its functional objectives while taking into consideration various non-functional requirements. This framework considers (a) the upstream factors (i.e. the "whys" traced to enterprise business objectives) that should be considered in the design of enterprise business processes, and (b) the downstream effect (i.e. the "hows") of abstract software systems and process architecture design and usage, including considering the interplays and trade-offs between them, such as designing for reusability, flexibility, repeatability, customizability etc.

3 Requirements for Process Architecture Modeling and Analysis

The diverse range of factors influencing and governing software-enabled enterprise transformation can be abstracted out as a set of emerging requirements for a conceptual modeling framework. These factors were identified through a study of literature on recent trends in enterprise transformation that result from recent advances in software technology. Stating the requirements in such a manner guides the development of the framework that would allow enterprise architects to study change in a structured and systematic manner as per each enterprise's individual needs.

An Architecture of Processes: Every organization relies on many processes that together ensure its success and viability. Different types of processes may take place over different timescales and have different frequencies of occurrence and execution. Enterprise modeling techniques need to express and reason about the nature of the relationships amongst the various business and software processes and their resultant artifacts.

Multiples Types and Levels of Processes: As process architectures encompass multiple individual processes, there can exist multiple levels of process dynamics within a process architecture. Through "levels", we indicate different types of processes that exist in the enterprise; some which are responsible for operational execution, others are responsible for strategy planning. There may even be processes that design additional processes. Boundaries would exist between these processes with activities on either side of the process boundary having certain attributes and behavior. These multiple levels of dynamics are not entirely evident to the casual observer nor are the boundary transitions (between these process levels) apparent. A process architecture would have to incorporate details and attributes that would allow for the identification of and differentiation between various processes.

Upfront vs. Deferred Planning: Processes modeling techniques generally describe process activities that are to be executed, but not how these process activities are determined. Such a consideration is important for enterprises undergoing change. Different plans may be prepared based on how they are to be executed by downstream process activities. Some plan-related activities may be left for later because of unavailability of updated contextual data, or to ensure some flexibility in process or system design.

Pushing Design Decisions Downstream: There exist two levels of processes, one where the process is responsible for the creation of a tool, capability or artifact while the other being responsible for (repeatedly) using the designed artifact. Different designs may be prepared based on how they are to be used by downstream process activities. Some design decisions would be deferred to at runtime (or use-time) to ensure some flexibility in the use of the design artifact.

Enterprise and Process Goals: Enterprises have defined functional and non-functional objectives and goals, with mechanisms and processes assigned to attain them. Any reconfiguration exercise to organizational processes would need to adhere to existing functional objectives while considering trade-offs amongst non-functional objectives. These objectives can be viewed as functional and non-functional requirements.

Feedback and Feedfoward Paths: The enterprise needs to "observe" and be aware of evolving situations, based on which it would initiate and undertake activities of transformation and change. Such paths of change can be analyzed in terms of *sense-and-response* loops through which the enterprise continuously adapts and improves [2]. Sensing and responding take place in business and technology processes that exist at different levels of dynamics and timescales.

Multiple Perspectives and Trade-Off Analysis: In many cases, there are multiple possible reconfigurations that need to be confirmed against other perspectives, and any consequence to enterprise goals and objectives. Trade-off analysis would need to be done to consider the impact of and to various systems-, enterprise-, process-, and social-level factors.

Represent and Reason about Speed, Timescales, and Process Cycles: Software-enabled enterprise transformation allow for increased enterprise benefits, such as like automation, higher productivity, greater release cadence etc. These often come at the expense of other enterprise considerations, like increase in complexity of solution, higher cost, reduced flexibility, reluctance in adoption, etc., which need to be considered.

4 The hiBPM Framework

Adaptable business processes and systems are becoming essential for modern enterprises who need to undergo continuous transformation. Thus, enterprises need to be designed for flexibility and modifiability. As techniques for implementing flexibility become well developed, enterprise architects and system designers face choices as to when and where to deploy what kinds of flexibilities in the enterprise. The hiBPM framework was originally introduced in [3] for providing a systematic and structure approach to determine points of flexibilities (or variations) in the overall collection of business processes that can be used to progressively reason about and introduce changes in enterprises, while ensuring that the enterprise continues to attain its organization objectives. As part of this PhD research study, we have enhanced this modeling framework through the elaboration of the various structural and relational elements, while introducing several new notions for managing enterprise change and integrating abstract software systems design into hiBPM.

In this section, we discuss some aspects of the hiBPM framework.

Architecting Processes

Several business processes are shown as part of a hiBPM process architecture model (or simply the hiBPM model) in Fig. 1 from a Loan Application domain example. These processes collectively depict activities, their relationships and interactions needed to accomplish process objectives, while highlighting architectural relationship between business process segments for enabling redesign. hiBPM proposes abstraction and reasoning to facilitate design decisions and emphasizes the existence of various decision-making points and architectural relationships between process segments while abstracting away from process-level details. The idea is to have sufficient expressiveness of the domain for allowing capturing and evaluation of alternative configuration options.

Accommodating Uncertainty. Understanding where and when options should be kept open is a central mechanism in the analyzing of process architecture. Alterative designs are different ways of respectively modifying or implementing the process architecture. A *variation point* (VP) has an associated objective and several *variants* for achieving them. By identifying and focusing on VPs, possible alternatives to process architecture design in the overall domain setting can be identified.

Deciding Between Variants using Goal Models. Goal models [4] provide the means to understand, analyze and guide possible configurations in the hiBPM model that help satisfy both functional and non-functional objectives. This is done through navigating the goal graph and seeing how a goal structure can be applied to an appropriate configuration of the hiBPM model.

Binding Variants. A fundamental idea for accommodating uncertainty is to keep options open. Determining where and when options should be kept open is therefore a central mechanism for process architecting. A VP is bound when one of the variants is selected after a suitable reasoning and analysis at that location is performed. Determining suitable variants, including when and where a VP is binded to a variant, is important during the analysis of hiBPM models.

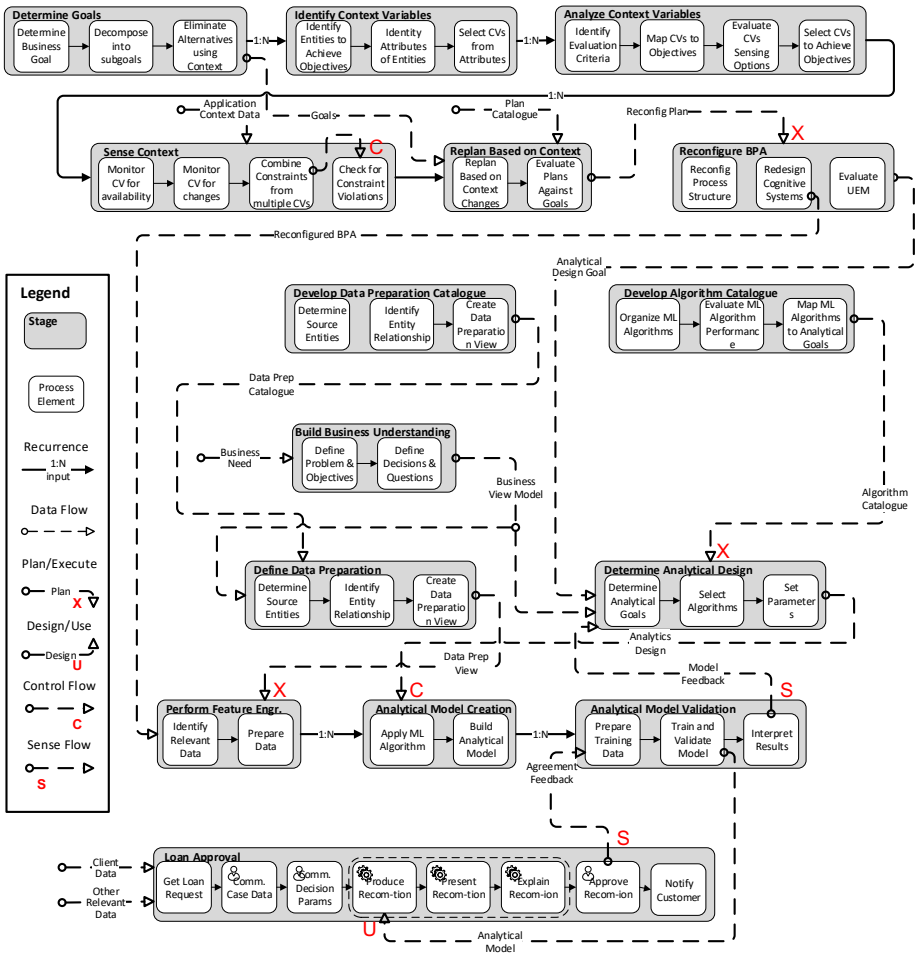

Fig. 1. hiBPM model for a Loan Application domain example

Process Elements

Process Elements (PEs) are basic process activity units which produce some output or outcome based on a set of control and data inputs. They may also include the act of making decisions as part of the activity processing. The specifics of how these are ac-

tions are performed are not necessary as the focus in hiBPM is on how the PE contributes towards the attainment of enterprise functions, and the relative positioning of the PEs within the overall process architecture.

Process Stages

A Process Stage (PS) is a collection of PEs that are to be executed collectively as part of the same execution cycle, while being generally structured in a manner where they deliver some enterprise objective in the form of functional or non-functional goals. A PS represents a (sub-)process within a broader business process that attains a defined business objective. Decisions and actions need to be identified that can be reused for multiple process instances, these (represented as PEs) can be put into a PS, thus saving time, money and possibly other resources. Another heuristic for creating PSes is to identify PEs that need to be executed with the same frequency or are triggered by the same data-driven trigger. As the emphasis of the hiBPM is on architectural relationships between business processes, the internals of PSes are usually only defined at a level where they show sufficient details on how the functional requirements are attained.

Temporal Variability

There are multiple possible temporal placements for PEs that achieve the same functional objective but are different in terms of their non-functional characteristics. A PE could be moved earlier or later in relation to other PEs, while being within the same PS. The output of the PS would not change, however the manner in which the PS is executed would change. Alternatively, a PE may move amongst PSes by either placing it in an upstream PS or a downstream PS. The output of the PS would change, with the purpose being to induce a change in how functional or non-functional goals are met. *Advancing* a PE relative to other PE reduces complexity and cost, as less effort is required to process the limited contextual information available at that instant. Advancement provides for stability and uniformity. Conversely, *postponing* a PE provides the benefit of executing it with the latest context and information available, thus reducing the risk and uncertainty that are inherent in any software process. Postponing PEs is done with the expectation that there will be better, more precise information available at some later point, which would allow for better, more context-sensitive outcomes.

Data Flows

Data Flow (or simply Flow) relationships are a means to transfer information objects from one PS to another. These information objects could take the form of data that are required by the downstream PS for its processing. Hence, the output of an upstream PS is transferred as an input to a downstream PS through these flow relationships. Flow relationships are an elementary construct in the hiBPM model and provide a simple hierarchical association between PSes. They provide some indication of the sequential execution of PSes in the overall process architecture and help with readability of the model.

Recurrence Relationships

Recurrence relationships between two PSes may have a recurrence attribute associated with them which indicates the relative execution frequency of PSes at both sides of the relationship. This execution frequency between two adjacent PSes can change relative to each other. A typical configuration is where, the downstream PS executes at a higher frequency than an upstream PS, which allows the downstream PS to use the same process input but with updated context. In some cases, the downstream PS can execute at a lower frequency than an upstream PS; this may be in conditions when the cost of process execution is high or rarely required. In the recurrence perspective, the choice is about in which PS to place a given PE in to achieve the right balance of reuse and flexibility. A PE can be moved from a PS with a lower recurrence to one with a higher recurrence (and vice versa). Such a movement of the PE can change the non-functional properties of the PS in various ways. For example, reducing the PE recurrence saves cost as the same PE does not have to be executed repeatedly. Conversely, increasing the PE recurrence can assist with flexibility and adaptability as the PE is executed based on updated and current information.

Trigger Relationships

Trigger relationships are used to indicate that the completion of an upstream PS (or a PE) will result in the immediate execution of an immediately downstream PS. This represents some form of temporal relationship between these two associated PSes. Triggers help with the relaying of change-related information through feedforward paths and may transcend different PSes of the overall process architecture, meaning that they may span regions where PSes can have different execution cycles, separate sets of responsibilities and functions, etc. They are initiators of different kinds of change across the process architecture, which can be realized in the form of process reconfiguration or system reconfiguration. Triggering conditions pertain to changes in context and their availability, thus detected changes in context allow for reconfiguration in other parts of the process architecture. Triggering a PS results in its re-executed.

Design-Use Relationships

In the hiBPM framework, business process can result in the creation of a tool, capability or artifact that can be used, referred to as a *design*, repeatedly by other enterprise business processes for attaining some process or enterprise goal. A Design-Use relationship exists between two PSes, with the PS creating the artifact called the design PS and the one using the artifact called the use PS. Introducing the Design-Use relationship allows modeling and analyzing changes in capabilities in continuously evolving systems. The assumption is that the design PS will not be executed just once to produce a tool or a capability. Rather, driven by changing business needs and external environments, as well as based on the feedback from the use of the current version of the tool, the design PS can be re-executed when appropriate. This will produce/acquire new versions of the capability, thus evolving the enterprise and/or its systems.

Flexibility in Design-Use. Flexibility designs have usage that vary differently in order to result in different business outcomes. These can be considered as evolvable objects which can be easily redesigned / modified at usage. The more single-purpose

(less flexible) the tool is, the simpler it is to use and the more optimized it can be. For a more flexible design (for supporting usage-time modifiability of the design), the design complexity may increase resulting in additional process overhead.

Evolving Capabilities. Design-Use variability is attained by positioning a PE on the design side or the usage side of a process, i.e., whether the PE is invoked as part of a design process, or is invoked during the usage of that artifact, tool, or capability that is the outcome of the design. In case of an activity, this means that the tool takes on more functionality, thus increasing the level of automation in the use PS. In case of a decision, it means that it is bound in the design PS and becomes fixed in the use PS, thus reducing the customizability of the produced tool.

Plan-Execute Relationships

A *plan* provides instructions for execution of process activities to accomplish enterprise functional and non-functional goals while simultaneously reducing the space of possible process execution possibilities, as there may be several possibilities to attain these enterprise goals. A plan is the output of the planning PS and can be an instruction set, an arrangement of actions, or a set of specifications that describes the method, means and constraints of executing the plan. Downstream PSes need to be aware of the instructions as codified in the plan in order to ensure proper execution. The relationship between the planning PS and the execution PS is called Plan-Execute.

Flexibility of Process Execution. Plan-Execute relationships supports the identification and analysis on variations of the completeness of plans being produced. A primary focus is on analyzing how much to pre-plan in the planning PS and how much to leave to the execution PS to achieve a desired level of flexibility in an organization. A PE can move from an execution PS to a planning PS (and vice versa) based on an analysis of their contribution to the relevant non-functional objective. Such movements create variations in the Plan-Execute behavior and allow for either increased pre-planning (by moving a PE to the planning PS) or shifting more responsibility to the execution side (by moving a PE to the execution PS). A plan produced by a PS either fully specifies or partially constrains the behaviour of the subsequent PSes. We refer to the former as complete plans and the latter as partial plans.

Feedback Paths

Feedback paths are used for revealing special adaptation relationships between two PSes. The visual representation in hiBPM models supports the analysis of feedback paths e.g., whether the adaptation loop is designed properly, where to source the data flow for feedback, and where to insert the feedback flow back into a higher-order PS. The change is either through a “control” flow constraining the possible options for the target process at runtime or through an “execute” flow that changes the space of options for its target process by creating new capabilities. Hence, there is a hierarchy among processes that reflects their relative control order. The execution frequencies of both these levels typically differ as well. To make feedback loops explicit, message flow that serves a Sense purpose are marked with an “S” on an output from a PS where Control is represented as a message flow adorned with “C” to mark a message flow that serves

as a control input to a PS. Mechanisms indicate a capability output that supports a PS. In hiBPM, we indicate mechanisms as designs, as mechanisms are to be used by downstream PSes.

Feedback Path Variability. Feedback paths variability can take one of two forms. First, through changing the adaptation loops so that sensing and controlling happen within an area of same execution frequency. Second, additional PSes may be modified as part of the adaptation consideration leading to more complex models that better handle the adaptation by performing additional planning, designing or increasing the execution frequency of PSes.

Human-Systems Relationship

Within any process architecture, there would be systems integrated in the business process which are utilized by human users for process execution or decision making. A human-systems relationship represents the elementary interactions that comprise engagements between users and information systems. These engagements are configured in a particular manner based on enterprise requirements, business domain constrains, the level of trust of human decision makers on those systems, the quality and availability of relevant data, and contexts. The study of this relationship is necessary as, within the enterprise, users are engaging with the new types of systems being introduced while dealing with real-world business situations. Both sides (i.e., the users and the systems) would need to adapt and adjust to each other and eventually converge to a workable state; the users learning to execute their assigned business processes while the systems undergo cycles of iterative improvements to make them significantly more efficient and intelligent. Enterprise information systems should be capable of supporting a variety of enterprise business process configurations, with the roles of these enterprise systems ranging from assistive, to advisory, to complete responsibility for decision making.

5 Related Work

Process architectures are used to provide abstract representation of multiple processes that exist in an enterprise [5]. Additional processes relationships are proposed in [6]. Process architectures can also be seen as a means for developing a more holistic view of the organization by associating business process modeling and enterprise architecture, while additionally decomposing processes into higher level of granularity [7]. Our notion of process architecture differs from these as we are focused on the need for ongoing change in the enterprise and use process architectures to model those changes by providing necessary structural and relational elements. Through such a representation of the architecture of business processes in the enterprise, we can analyze possible variants of process architecture configurations that exist, including how enterprise functional and non-functional goals are satisfied. The concept of business process architecture also exists in the enterprise architecture area. For example, in ArchiMate, business process cooperation includes causal relationships between business processes,

mapping of business processes onto business functions, realization of services by business processes, and the use of shared data [8], and can also imply the type of relationships among business processes.

Traditional business process modeling notations, such as BPMN [9], rely on an imperative approach where the process model represents the process state of the system and all permitted actions. Declarative process modeling notation (such as BPMN-D) allows the capturing of constraints on activity flows [10]; any flow is permitted as along as the constraints are upheld. Other approaches in BPM have focused on the role of "artifacts" within process design and execution; the argument being made that without having an understanding of the information context, business participants often are too focused on execution of process activities without understanding the reasons for the execution, thus limiting opportunities for operational efficiency and process innovation [11]. Here a business artifact is self-contained, trackable "instance of a flow entity in the network" and has a unique identity. Flexibility in the design of Process Aware Information Systems (PAIS) is needed for dealing with design-time uncertainties and managing evolving business processes [12].

6 Research Approach

Design Science Research

Design Science Research (DSR) is a research methodology well-suited for Information Systems research due to the inclusion of social and organizational aspects as part of the research process [13]. DSR is more agreeable towards the development of technology-based or technology-derived artifacts, such as conceptual modeling frameworks and software tools, along with their empirical evaluation in an actual study environment. The desired research outcome stated previously strongly correlates to the design artifacts introduced in DSR. The guidelines-based design science research approach proposed in [13] was used in this PhD project for the development and validation of design artifacts pertaining to the proposed research framework. Here the design artifacts include the hiBPM conceptual modeling framework that consists of a set of process-based constructs and accompanying methods that help in modeling and analyzing the process architecture design.

Case Study

We supplemented the DSR overarching approach with confirmatory case studies [14] for the evaluation, validation and ongoing refinement of the developed conceptual modeling framework in an actual enterprise environment. Two organizations were approached in order to be able to generalize findings and observations across a range of circumstances, while refuting theories and ideas which may be valid in limited circumstances and localized settings. These two case studies thus ensured greater validity, generalizability, applicability and rigor of the research exercise, while reducing researcher bias and interpretability of data collection and analysis. Through an active en-

gagement spanning multiple months, the hiBPM framework was applied to both organizations' unique setting for validating if the framework provided sufficient expressiveness for allowing for the evaluation of different process architecture configurations that satisfy the non-functional requirements under evolving circumstances.

7 Research Contribution

This research yielded both theoretic and practical benefits through advancing current conceptual modeling techniques for understanding, evaluating and analyzing software-enabled enterprise transformation activities. This was achieved by enhancing the hiBPM framework through improved elaboration of the various structural and relational elements for addressing practices such as continuous enterprise change, using concepts of multi-level business processes, and linking process architecture design considerations with software systems integration, organizational stakeholder interests and enterprise-level business goals. Basic transformation types and dimensions were better established, along with methods for their implementation in an enterprise context by focusing on involved processes and artifacts. The concept of feedback and feedforward loops was clearly specified which, along with methods for external context identification and handling, were able to help with dealing ongoing and iterative enterprise change. Further, methods for assessing, reasoning and selecting suitable alternatives are provided while considering trade-offs with regards to enterprise goals. Such a framework is a step forward and it allows for the structured contemplation of enterprise transformation, brought about by changes in business processes and software technologies.

Acknowledgements
I would like to recognize my PhD supervisor, Prof. Eric Yu, for his ongoing support as I progress through the doctoral program and establish a research career for myself.

References

1. Wilkinson, M.: Designing an "adaptive" enterprise architecture. BT Technology Journal, 24(4), pp. 81-92 (2006)
2. Haeckel, S.: Adaptive enterprise: Creating and leading sense-and-respond organizations. Harvard Business Press (2013)
3. Lapouchnian, A., Yu, E., Sturm, A.: Re-designing process architectures towards a framework of design dimensions. In International Conference on RCIS, pp. 205-210, IEEE (2015)
4. Van Lamsweerde, A.: Goal-oriented requirements engineering: A guided tour." In Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on, pp. 249-262, IEEE (2001)
5. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.: Fundamentals of Business Process Management, Ch.2. Springer-Verlag, Berlin-Heidelberg (2013)

6. Eid-Sabbagh, R., Dijkman, R., Weske, M.: Business process architecture: use and correctness. In Proc. 10th International Conference on Business Process Management (BPM'12), pp. 65-81, Springer (2012)
7. Malinova, M., Leopold, H., Mendling, J.: An empirical investigation on the design of process architectures. In 11th International Conference on Wirtschaftsinformatik, pp. 1197-1211, (2013)
8. ArchiMate 3.0 Specification. Retrieved from http://pubs.opengroup.org/architecture/archimate3-doc/
9. Business process Model and Notation, v2.0,http://www.omg.org/spec/BPMN/2.0/PDF/
10. De Giacomo, G., Dumas, M., Maggi, F. M., & Montali, M.: Declarative process modeling in BPMN. In International Conference on Advanced Information Systems Engineering, pp. 84-100, Springer, Cham, (2015)
11. Bhattacharya, K., Gerede, C., Hull, R., Liu, R., & Su, J.: Towards formal analysis of artifact centric business process models. In International Conference on Business Process Management, pp. 288-304, Springer Berlin Heidelberg, (2007)
12. Reichert, M., & Weber, B.: Enabling flexibility in process-aware information systems: challenges, methods, technologies. Springer Science & Business Media (2012)
13. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design Science in Information Systems Research, MIS Quarterly, 28, 1, pp. 75-105 (2004)
14. Easterbrook, S., Singer, J., Storey, M. A., Damian, D.: Selecting empirical methods for software engineering research. In Guide to Advanced Empirical Software Engineering, pp. 285-311, Springer London (2008)