A Goal-Oriented Requirements Engineering Method for Business Processes

Ken Decreus, Geert Poels

Faculty of Economics and Business Administration, Ghent University, Belgium.
{ken.decreus | geert.poels}@ugent.be

Abstract. Central to the development of BPMS technology was the promotion of a new language, Business Process Modelling Notation (BPMN). The primary goal of BPMN is to provide a common language for describing process behaviour, shareable by business and IT, which includes business users, business analysts, and technical developers. What seems to be missing in the way that business users are supposed to use BPMN, is an explicit consideration of the strategic rationale of having certain business processes as well as support for describing business processes in terms familiar to business people. We extended current work on Goal-Oriented Requirements Engineering (GORE) for business process design, i.e. B-SCP framework [1] and the work of Lapouchnian et al. [2], in order to obtain an appropriate GORE for BPMN modelling method. Our first contribution is the introduction of a B-SCP metamodel, which has been implemented by means of the Eclipse Modelling Framework. Our second contribution is an Eclipse-based B-SCP editor that enables business users to specify their strategic requirements and operational tasks. Our third contribution consists of model transformations to generate BPMN skeletons out of the B-SCP model, which were implemented by means of the Atlas Transformation Language.

Keywords: Goal-Oriented Requirements Engineering, Business Process Modelling, Business-Strategy Context Process, Atlas Transformation Language

1 Introduction

In the last three decades, an increasing attention to business process change as a factor of organizational success has been witnessed. In particular from 2000 onwards, Business Process Management (BPM) technologies have gained world-wide popularity [3]. One of the main BPM-enabling technologies is the Business Process Management System (BPMS), which Smith and Fingar [4] define as a modelling, integration, and execution environment for the design, manufacture and maintenance of business processes. The importance of BPMS is illustrated by Gartner [5], who predicts that by 2015 30% of business applications will be developed by means of BPMS technology.

Central to the development of BPMS technology was the promotion of a new language, Business Process Modelling Notation (BPMN), which could be used to represent business processes. As given by the BPMN specification [6], the primary goal of BPMN is "to provide a notation that is readily *understandable* by all business

users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes." (p1, [6]). Silver stresses the importance of using BPMN as a *common language* between business and IT. Furthermore, Silver [7] distinguishes different types of BPMN modelling, depending on the user category. Firstly, BPMN Level 1, or *descriptive* modelling, is geared towards the business user and offers a basic set of BPMN elements. Secondly, BPMN Level 2, or *analytical* modelling, supports the business analyst in using the complete BPMN notation to describe the activity flow precisely, including the exception paths. These models should be complete and consistent, but not yet contain technical details to make them executable. Thirdly, BPMN Level 3, or *executable* modelling, allows the technical developers to add process data, service interfaces and human task assignment that are needed to execute the BPMN models using BPMS technology.

When we look at BPMN Level 1, the business user is already expected to understand and work with BPMN concepts such as pool, lane, task, subprocess, start event, stop event, exclusive gateways, parallel gateways, sequence flow, and message flow, which are terms that, maybe apart from task, do not belong to the ordinary language used by business people. It is doubtful whether business people (e.g. accountants, marketers, sales people, auditors, finance officers, stock managers, etc.) think of business processes in terms of 'lanes', 'pools', 'gateways', and 'events'. Havey [8] warns that BPMN is not suited for business users, and stresses the importance of capturing requirements based on an approach that business users can understand. Fernandez et al. [9] confirm this finding and state that BPMN scores low on usability for business users.

Business managers also frequently need to deal with complex real-world problems (e.g. how to react to the entry of a new, low-cost service provider in the market?) that require considering simultaneously high-level strategic requirements and low-level operational details. However, Recker [10] states that BPMN currently lacks concepts to support process decomposition and organisational modelling. Recker [10] suggests to use a different, easier, and more business user adapted approach to process modelling with BPMN, by providing dedicated symbols for placing a process into its organisational and hierarchical context.

What seems to be missing in BPMN Level 1, i.e., the way that business users are supposed to use BPMN, is an explicit consideration of the strategic rationale of having certain business processes as well as support for describing business processes in terms familiar to business people. In attempting to deal with this matter, this paper addresses the following research question:

RQ: How can business users design complex business processes in terms of and in correspondence with strategic requirements?

To answer this research question, we developed a new BPMN approach to business process modelling targeted at business users (i.e., BPMN level 1 as referred to by Silver [7]). We assume that the context of our approach consists of a real-world environment in which there is a strategic interest of business users in the design of the business processes. As Wieringa and Heerkens [11] explain, the solution design phase

proposes an improvement to a problematic situation, and is based on specific solution properties. In our approach, the main solution properties are:

(i) consideration of the strategic rationale of having certain business processes, and having them organized in certain ways
(ii) support for describing business processes in terms familiar to business users and linking these business processes to strategic requirements

This paper is structured as follows. Section 2 provides details on the background of our work. Section 3 shows an overview of our approach and introduces the full implementation details of it. Section 4 offers a discussion about our approach. Section 5 concludes this paper and introduces future work.

2 Background

Our approach heavily relies on previous Goal-Oriented Requirements Engineering (GORE) research, which aimed at developing ways to capture high-level strategic business requirements and use them to drive the system development process. To this end, we investigated [12] the GORE and BPM literature to find studies that apply goal-oriented requirements engineering to business process design. We found that methods that apply GORE techniques for business process modelling, generally lack clear mappings between goal concepts and business process concepts and are short of detailed transformation descriptions (for details of this study, see [12]). Therefore, we were not able to reuse their transformations in our research. Some methods, however, provide a sound basis on which we can build our approach, i.e., the B-SCP framework [1] and the work of Lapouchnian et al. [2], which we will briefly introduce in this section.

To start with, the B-SCP framework [1] is a requirements engineering framework for organizational IT that directly addresses an organization's business strategy and the alignment of IT requirements with that strategy. Goal modelling is used to represent business strategy as requirements, and Jackson context diagrams [13] to represent business and system model context. The strategy and context parts are integrated using a problem diagram framework [13]. Strategy is first elicited using VMOST [14], an organizational alignment analysis technique. Then, an i* goal model [15] is constructed using goal modelling rules for organizational motivation proposed by OMG's Business Motivation Model [16]. To refine requirements from a strategic, high-level problem diagram down to the lowest operational level, a progression of problem diagrams is used to represent this top-down hierarchy. In addition, the combined goal and problem diagrams are briefly mapped to Role Activity Diagrams (RAD) [17], but we did not consider these mappings due to our earlier findings [12].

Next, Lapouchnian et al. [2] propose a requirements-driven method for configuration of high-variability business processes in terms of business priorities. This method is characterized by textual annotations to add control flow detail to goal models, which we will reuse in this paper. For instance, the sequence annotation (";") can be added to AND decomposition to indicate that all the subgoals are to be achieved in sequence from left to right. As we aim at BPMN Level 1 [7], we only consider annotation of sequential AND decomposition, parallel AND decomposition,

and OR decomposition. The annotation of control flow is organised per group of decomposed requirements (e.g. all subrequirements of one requirement have a sequential AND decomposition), so it is impossible to have different kinds of control flow annotations in the same group of requirements.

3 Overview of our approach

Our approach to BPMN business process modelling for business users consists of four steps. First, the business user applies the original B-SCP method [1, 18] and uses our visual editor to create a B-SCP model. Secondly, the business user decides to elaborate the process aspects of a specific part of the B-SCP model, by adding control flow annotations [2] that are needed for BPMN model generation. Thirdly, the business user uses the computer-based model transformations to generate BPMN process model skeletons, and finally, the business analyst takes the BPMN process model skeleton as input for his work and creates a consistent and complete BPMN business process diagram (compatible with BPMS technology).

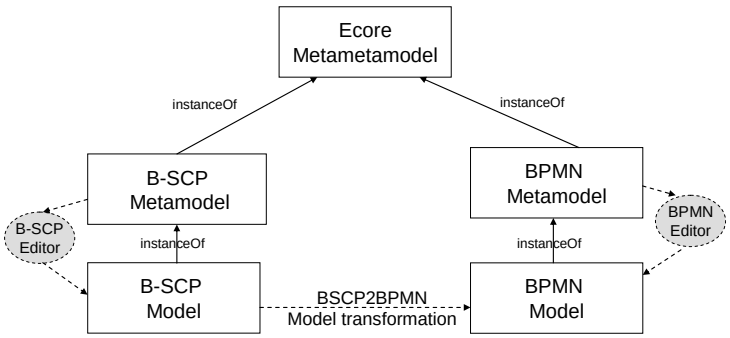

Figure 1: Overview of GORE for BPMN

To realize our approach, a layered implementation architecture (Figure 1) has been developed in IBM's Eclipse environment [19]. The fundaments of our solution are built upon the different abstraction layers of the OMG Model Driven Architecture [20]. On top, the high-level metametamodel in Ecore (e.g. defining elementary constructs like Class and Relationship) is used to define medium-level metamodels (e.g. containing the instance of a Class called Goal), of which models are defined on the lowest-level (e.g. containing the instance of a Goal called 'Shorten Cash Cycle'). In this paper, we use two different medium-level metamodels, i.e., one metamodel to define strategic business requirements and context (B-SCP) and another metamodel to represent business processes (BPMN). Both metamodels have associated tool support to allow users to visually edit model instances.

The properties of our approach are supported by our implementation as follows. The consideration of the strategic rationale of having certain business processes, and having them organized in certain ways, is supported by the our B-SCP metamodel and corresponding B-SCP editor to create B-SCP models. The support for describing business processes in terms familiar to business users and linking these business

processes to strategic requirements is given by the annotation of control flow via the B-SCP editor and by means of the BSCP2BPMN model transformations.

Section 2.1 introduces B-SCP terminology and explains how we created the B-SCP metamodel, Section 2.2 shows the B-SCP editor and clarifies how business users add control flow annotations to B-SCP models in the B-SCP editor, and Section 2.3 provides insights into the model transformations from B-SCP to BPMN. To illustrate the concepts, we use a fictive car rental company (EU-Rent) as running example.

3.1 B-SCP Metamodel

The syntax of a language is determined by the set of symbols that compose the language as well as the rules for forming valid combinations of these symbols [21]. The original work on B-SCP [1, 18] defines a set of symbols based on the i* goal language [15] and Jackson problem frames [13], and informally explains the rules for forming valid combinations of these symbols. Our work introduces a B-SCP metamodel (Figure 2) to define the abstract syntax of the B-SCP language.

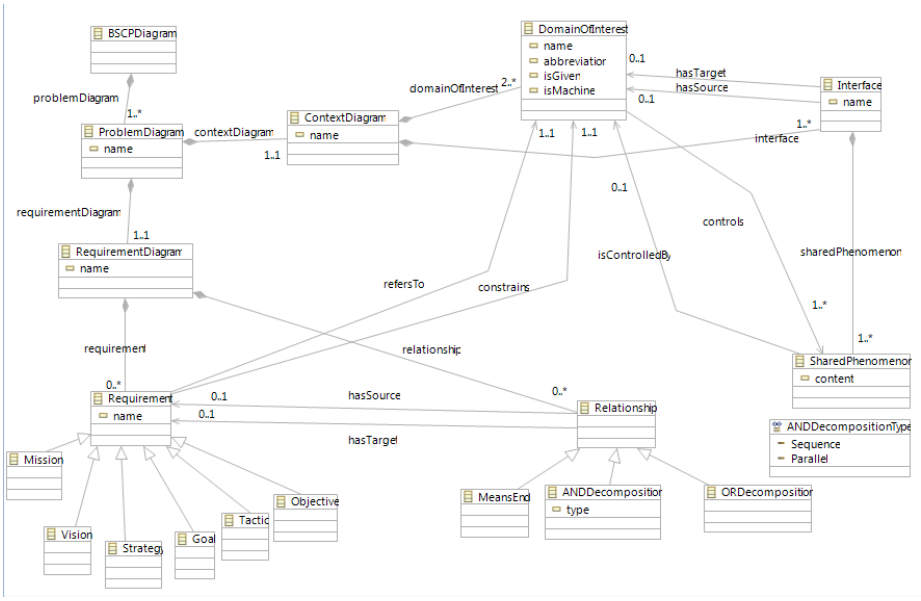

Figure 2: B-SCP Metamodel in Ecore

A *BSCPDiagram* contains one or more *ProblemDiagrams*, which have each exactly one *RequirementDiagram* and *ContextDiagram*. A *ProblemDiagram* may refine elements of another *ProblemDiagram*, which makes a *BSCPDiagram* an hierarchical structure of *ProblemDiagrams*.

A *RequirementDiagram* can contain many *Requirements*, where *Requirement* is a generalization of Mission, *Vision*, *Strategy*, *Goal*, *Tactic* and *Objective*. As defined by the OMG's Business Motivation Model [16], a *Vision* describes the future state of the enterprise, without regard to how it is to be achieved, and *Mission* indicates the

ongoing activity that makes the vision a reality. For instance, EU-Rent could have a vision to 'Be the car rental brand of choice for business users', and a mission to 'Provide car rental service across Europe for both business and personal customers'. Next, a *Goal* indicates what must be satisfied on a continuing basis to effectively attain the vision, and a *Strategy* is a long-term activity designed to achieve a goal. For instance, the goal 'Be a premium brand car rental company' tries to attain EU-Rent's vision, and a strategy 'Target major airports to find business users' supports the achievement of the EU-Rents' goals. Finally, an *Objective* is a specific and measurable statement of intent whose achievement supports a goal, and a *Tactic* is a short-term action designed to achieve an objective. For instance, the objective 'Be rated by AC Nielsen in top 6 car rental companies' supports the EU-Rent's goal to be a premium brand, and the tactic 'Encourage rental extensions' would be a short-term action to score better in listings such as AC Nielsen.

Requirements described in *RequirementDiagrams* are interconnected via *Relationships*, such as *MeansEnd*, *ORDecomposition,* and *ANDDecomposition*. A *MeansEnd* link indicates a relationship between an end and a means for attaining it [1]. For instance, the vision 'Be the car rental brand of choice for business users' is an end supported by its mission 'Provide car rental service across Europe for both business and personal customers' as means. Next, an *ORDecomposition* link indicates that a requirement is fulfilled if at least one of the lower-level requirements are fulfilled [1]. For instance, a tactic 'Handle Rental Extensions' could be fulfilled by lower-level tactics such as 'Use own staff to extend rental' or 'Use airport staff to extend rental'. Finally, an *ANDDecomposition* link indicates that a requirement is fulfilled if all lower-level requirements are fulfilled [1]. In this paper, we distinguish between sequential and parallel fulfilment of *ANDDecomposition* links. For instance, the tactic 'Encourage rental extensions' can be decomposed into two sequential tactics, of which 'Persuade airport customers' is the first in time and 'Handle rental extensions' is the second. In contrast, the tactic 'Persuade airport customers' might be decomposed into parallel tactics that can be executed simultaneously, such as 'Offer extra flight miles' and 'Offer free cabrio upgrade'.

A *ContextDiagram* contains at least two *DomainsOfInterest* and at least one *Interface* to connect a pair of *DomainsOfInterest*. For instance, *DomainOfInterest* EU-Rent has an interface with *DomainsOfInterest* business customer, personal customer and airport. An *Interface* should contain at least one *SharedPhenomenon* that is controlled by a specific *DomainOfInterest*. For instance, domains EU-Rent and airport might share phenomena such as airport location, welcoming of customer, or holiday season.

A domain of interest in the context diagram *describes* a part of the real-world, whereas a requirement *prescribes* the domain of interest in the context diagram. The connection between requirements and context is made by using the *refersTo* and *constrains* relations from a *Requirement* to a *DomainOfInterest*. For instance, the requirement 'Be the car rental brand of choice for business users' *refers to* domain EU-Rent, as this requirement involves the EU-Rent domain without constraining the way that EU-Rent becomes the car rental brand of choice. In contrast, the requirement 'Use own staff' *constrains* the domain EU-Rent Airport centre in making the staffing planning, as this requirement restricts the way that EU-Rent Airport organizes its staffing.

3.2 B-SCP Editor

In order to instantiate B-SCP models from the B-SCP metamodel, modellers need an intuitive and graphical environment. The *graphical modelling framework* [22] project takes an Ecore metamodel (such as our B-SCP metamodel) as an input and offers a step-by-step approach to generate a fully functional graphical editor (details can be found at [23]). In Figure 3, the running example of EU-Rent is visualised in the Eclipse-based B-SCP editor.

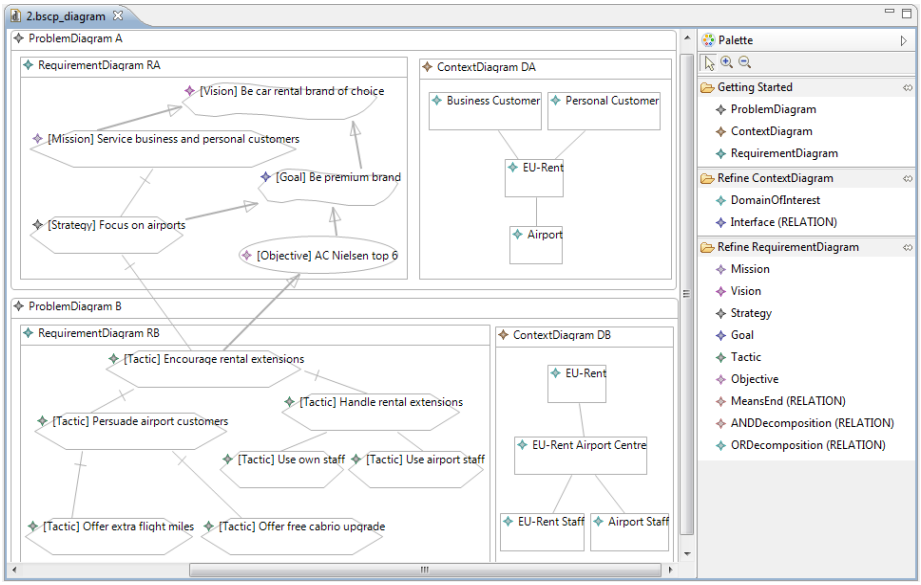

Figure 3: Eclipse-based Visual B-SCP Editor

At some point in the problem diagram hierarchy, operational details about the business processes are specified. When a business user creates a B-SCP *ProblemDiagram* that solely exists of tactics, we consider this problem diagram to represent (a part of) a business process, so it becomes useful to add control flow annotations. Although Lapouchnian et al. [2] recommend textual annotations to add control flow detail to requirement models, we choose to add such annotations via the *properties* pane of the B-SCP editor to lower the visual complexity of the models. For instance, Figure 4 shows an OR Decomposition, Figure 5 illustrates how and AND Decomposition is annotated with sequence, and Figure 6 displays the parallel annotation.

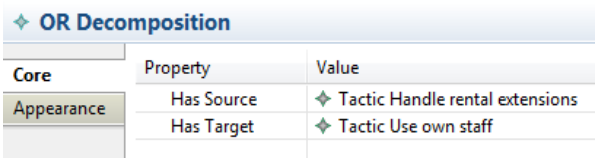

Figure 4: Using OR Decomposition

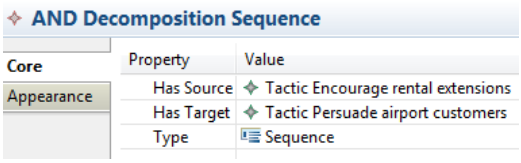

Figure 5: Setting AND Decomposition to Sequence

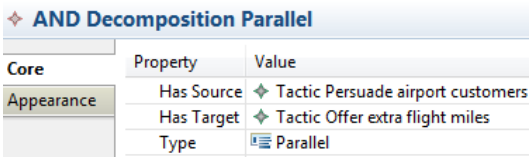

Figure 6: Setting AND Decomposition to Parallel

3.3 Model Transformation B-SCP to BPMN

When a B-SCP *ProblemDiagram* meets certain transformation criteria, our model transformations can be used to transform this diagram into the skeleton of a BPMN business process diagram that can be further refined by a business process analyst to achieve completeness and consistency. Our transformation criteria are as follows. The requirement diagram (related to the problem diagram to be transformed) should only contain tactics, the top tactic should represent a business process, the control flow annotations should be consistent per group of tactic decompositions, each tactic should refer to or constrain one domain of interest (of the context diagram), and there should be at least one shared phenomenon on each interface between domains of interest.

Next, we will elaborate on the B-SCP to BPMN concept mappings that we created, which are implemented by means of the *atlas transformation language* [24] (details can be found in Appendix A). In general, Rules 1 to 4 are used to transform the main concepts, Rules 5 to 9 relate to the control flow transformation, and Rule 10 takes care of the generation of message flows.

- Rule 1 transforms a top node in a *RequirementDiagram* (e.g., Figure 3 – *Encourage rental extensions*) into business process diagram (e.g., the diagram shown in Figure 7).
- Rule 2 transforms a domain of interest (e.g., Figure 3 – *EU-Rent*) into a pool, a start event, a sequence edge, and an end event (e.g., Figure 7 – Labelled with (2)).
- Rule 3 transforms a medium node (e.g., Figure 3 – *Persuade airport customers*) of a *RequirementDiagram* into a sub-process (e.g., Figure 7 – Labelled with (3)).
- Rule 4 transforms a leaf node (e.g., Figure 3 – *Offer extra flight miles*) of a requirement diagram into a task (e.g., Figure 7 – Labelled with (4)).
- Rule 5 transforms the first occurrence of an OR Decomposition (e.g., Figure 3 – Link between *Handle rental extensions* and *Use own staff*) into two Gateway

Data-Based Exclusive activities and two sequence edges (e.g., Figure 7 – Labelled with (5)).

- Rule 6 transforms the other occurrences of an OR Decomposition (e.g., Figure 3 – Link between *Handle rental extensions* and *Use airport staff*) into two sequence edges (e.g., Figure 7 – Labelled with (6)).
- Rule 7 transforms an AND Decomposition with sequence (e.g., Figure 3 – Link between *Encourage rental extensions* and *Persuade airport customers*) into two sequence edges (e.g., Figure 7 – Labelled with (7)).
- Rule 8 transforms the first occurrence of a parallel AND Decomposition (e.g., Figure 3 – Link between *Persuade airport customers* and *Offer extra flight miles*) into two Gateway Parallel activities and two sequence edges (e.g., Figure 7 – Labelled with (8)).
- Rule 9 transforms the other occurrences of parallel AND Decomposition (e.g., Figure 3 - link between *Persuade airport customers* and *Offer free cabrio upgrade*) into two sequence edges (e.g., Figure 7 – Labelled with (9)).
- Rule 10 transforms a shared phenomenon, between two domains of interest (e.g., Figure 3 – shared phenomenon x between *EU-Rent* and *EU-Rent Airport Centre*), into a message edge x with a 'send' and 'receive' task and two sequence edges (e.g., Figure 7 – Labelled with (10)).

Table 1: BSCP2BPMN Concept Mappings

Rule Nr	B-SCP Concept	BPMN Concept
1	Top node (in Requirement Diagram)	BPMN Diagram
2	Domain Of Interest (in Context Diagram)	Pool
		Start Event
		Sequence Edge
		End Event
3	Medium node (in Requirement Diagram)	SubProcess
4	Leaf node (in Requirement Diagram)	Task
5	OR Decomposition – first occurrence	2 x Gateway Data-Based Exclusive Activity
		2 x Sequence Edge
6	OR Decomposition – other occurrences	2 x Sequence Edge
7	AND Decomposition (Sequence)	Sequence Edge
8	AND Decomposition (Parallel) – first occurrence	2 x Gateway Parallel Activity
		2 x Sequence Edge
9	AND Decomposition (Parallel) – other occurrences	2 x Sequence Edge
10	Shared Phenomenon (between two domains of interest a and b)	'Send' Task (at first pool)
		Sequence Edge (at first pool)
		'Receive' Task (at second pool)
		Sequence Edge (at second pool)
		Messaging Edge from 'Send' to 'Receive' Task

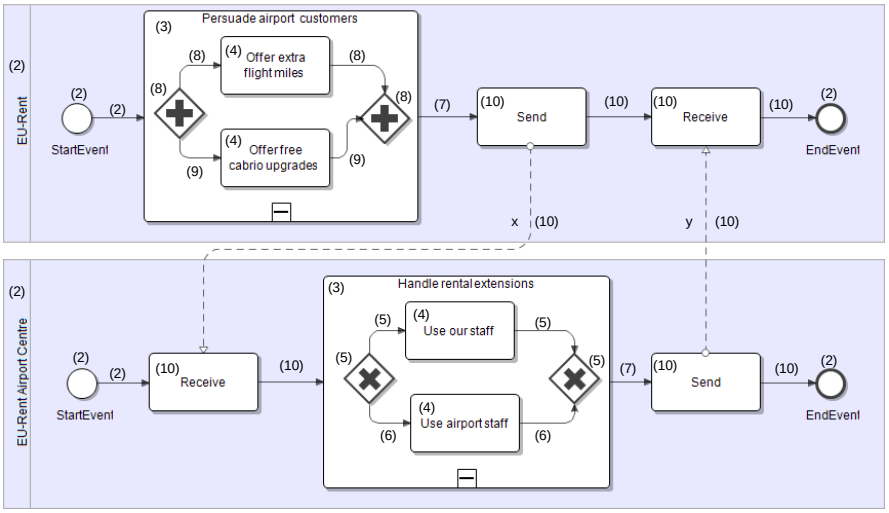

Figure 7: Resulting BPMN diagram for the B-SCP tactic 'Encourage rental extensions'

4 Discussion

4.1 Using B-SCP as Modelling Aid

Originally, B-SCP was proposed as requirements engineering framework for validating strategic alignment of organisational IT, based on manual interpretation of traceability links between strategy and technology. In this paper, we reused the work on B-SCP for a different purpose, that is as modelling aid for business users to design business processes linked to strategic requirements. We believe that B-SCP offers a well-documented and scalable alternative to the currently available modelling methods that combine strategic goals and business processes, which are often i*-based modelling languages [12]. Few published studies exist on applying the i* goal language into practice, and indications exists that practitioners of large-scale industrial projects are unable to understand i* models well enough to validate the requirements of the system they were building [25]. As the B-SCP framework was proposed to address the known shortcomings of i* and to leverage the existing knowledge of Jacksons's Problem Frames, we considered the B-SCP framework as the starting point of our work.

4.2 Separating Requirements and Business Processes

The differentiation between a goal-oriented requirements language and a business process language is the result of a deliberate design choice. As a modelling language is always conceived with a certain purpose in mind [26], we believe it is easier to

represent goals and business processes using different languages, and to provide model-based translations between these languages, instead of choosing one modelling language to represent both goals and business process concepts. With low modelling complexity (e.g. modelling one clearly understood business process), creating a requirements model could be seen as an overhead cost. But, as real-world business process modelling projects often quickly grow in complexity, business users can use a requirements model as an overview (or one could say, an overarching strategically aligned business process architecture), and generate as much business process models from the requirements model as they require.

5 Conclusion, Limitations and Future Work

Central to the development of BPMS technology was the promotion of a new language, Business Process Modelling Notation (BPMN). The primary goal of BPMN is to provide a common language for describing process behaviour, shareable by business and IT, which includes business users, business analysts, and technical developers. What seems to be missing in the way that business users are supposed to use BPMN, is an explicit consideration of the strategic rationale of having certain business processes as well as support for describing business processes in terms familiar to business people. This paper presents an approach that allows business users to design complex business processes in correspondence with strategic requirements. The main claim of this work is two-fold:

(i) The consideration of the strategic rationale of having certain business processes, and having them organized in certain ways, is important for a business user during BPMN modelling. By extending the work of Bleistein et al. [1], we created a B-SCP metamodel and offered the business user a graphical B-SCP editor (that corresponds to the B-SCP metamodel) to express the strategic rationale and the business processes related to the strategic rationale.
(ii) Support for describing business processes in terms familiar to business users and linking these business processes to strategic requirements. By extending the work of Lapouchnian et al. [2], business users can annotate control flow via the B-SCP editor. Then, specific parts of the B-SCP models can be transformed in corresponding BPMN skeletons by means of the BSCP2BPMN model transformations.

The main limitations of our approach is the lack of full-scale validation and the absence of the reverse transformation (from BPMN to B-SCP). Firstly, a full-scale validation is needed to evaluate the properties of our approach, and to investigate whether these properties contribute positively to BPMN modelling for business users. In order to tackle this shortcoming, we are in the process of applying the Seven-Eleven Japan [27] case exemplar to our approach to investigate the feasibility, and we are conducting case study research [28] at two organisations to discover the added value of our approach. Secondly, our work presents a top-down modelling method, which enables business users to transform parts of B-SCP models into BPMN skeletons, but the reverse transformation (from BPMN to B-SCP) is currently not

supported. To this end, our future work focuses on adding reference data (during the BSCP2BPMN transformation) into the BPMN models to allow reverse transformations to the B-SCP model, and the reverse BPMN2BSCP model transformations will be introduced.

References

1. Bleistein, S.J., Cox, K., Verner, J., Phalp, K.T.: B-SCP: A requirements analysis framework for validating strategic alignment of organizational IT based on strategy, context, and process. Information and Software Technology **48** (2006) 846-868
2. Lapouchnian, A., Yu, Y., Mylopoulos, J.: Requirements-Driven Design and Configuration Management of Business Processes. Business Process Management (2007) 246-261
3. Harmon, P.: Business Process Change, Second Edition: A Guide for Business Managers and BPM and Six Sigma Professionals. Morgan Kaufmann (2007)
4. Smith, H., Fingar, P.: Business Process Management (BPM): The Third Wave (2004)
5. Woods, J., Genovese, Y.: Delivery of Commodity Business Applications in a BPMS Does Not Mean You Should Customize the Applications. Gartner Research **G00139084** (2006)
6. OMG: BPMN 1.2 specification - http://www.omg.org/spec/BPMN/1.2/. (2009)
7. Silver, B.: BPMN Method and Style. Cody-Cassidy Press (2009)
8. Havey, M.: Keeping BPM simple for business users: power users beware. BPTrends **January** (2006)
9. Fernández, H.F., Palacios-González, E., García-Díaz, V., Pelayo G-Bustelo, B.C., Sanjuán Martínez, O., Cueva Lovelle, J.M.: SBPMN -- An easier business process modeling notation for business users. Computer Standards & Interfaces **32** (2010) 18-28
10. Recker, J.: Opportunities and constraints: the current struggle with BPMN. Business Process Management Journal **16** (2010) 181-201
11. Wieringa, R., Heerkens, J.: The methodological soundness of requirements engineering papers: a conceptual framework and two case studies. Requirements Engineering **11** (2006) 295-307
12. Decreus, K., Snoeck, M., Poels, G.: Practical Challenges for Methods Transforming i* Goal Models into Business Process Models. 17th IEEE International Requirements Engineering Conference, Atlanta (2009)
13. Jackson, M.: Problem Frames: Analysing and Structuring Software Development Problems. Addison-Wesley, New York (2000)
14. Sondhi, R.: Total Strategy. Airworthy Publications International Ltd (1999)
15. Yu, E.: Towards modelling and reasoning support for early-phase requirements engineering. Requirements Engineering, 1997., Proceedings of the Third IEEE International Symposium on (1997) 226-235
16. OMG: Business Motivation Model - http://www.omg.org/spec/BMM/. (2009)
17. Ould, M.A.: Business Processes: Modelling and Analysis for Reengineering and Improvement. Wiley, Chichester, New York (1995)
18. Bleistein, S.J., Cox, K., Verner, J.: Validating strategic alignment of organizational IT requirements using goal modeling and problem diagrams. JSS **79** (2006) 362-378
19. Eclipse: EMF - Eclipse Modelling Framework - http://www.eclipse.org/modeling/emf/.
20. OMG: Model Driven Architecture - http://www.omg.org/mda/. (2009)
21. Guizzardi, G.: On Ontology, ontologies, Conceptualizations, Modeling Languages, and (Meta)Models. Proceeding of the 2007 conference on Databases and Information Systems IV: Selected Papers from the Seventh International Baltic Conference DB&IS'2006. IOS Press (2007)

22. Eclipse: GMF - Graphical Modeling Framework - http://www.eclipse.org/modeling/gmf/.
23. Decreus, K.: Google Project Site - http://code.google.com/p/brm2bpmn/. (2010)
24. Eclipse: ATL - ATLAS Transformation Language - http://www.eclipse.org/m2m/atl/.
25. Maiden, N.A.M., Jones, S.V., Manning, S., Greenwood, J., Renou, L.: Model-Driven Requirements Engineering: Synchronising Models in an Air Traffic Management Case Study. Advanced Information Systems Engineering (2004) 3-21
26. Mylopoulos, J.: Information modeling in the time of the revolution. Information Systems **23** (1998) 127-155
27. Nagayama, K., Weill, P.: Seven Eleven Japan: Reinventing the Retail Business Model. CISR Working Paper 338 - MIT Sloan WP 4485-04 (2004)
28. Yin, R.K.: Case Study Research: Design and Methods, Fourth Edition, Vol. 5. SAGE Publications (2009)

Appendix A: BSCP2BPMN

```
(1) rule ProblemDiagram{
     from a : BRM!ProblemDiagram
     to b : BPMN!BpmnDiagram(name <- a.name)}

(2) rule DomainOfInterest{
     from a : BRM!DomainOfInterest
     to b : BPMN!Pool(name <- a.name),
        startevent : BPMN!Activity(activityType <- 'EventStartEmpty'),
        endevent : BPMN!Activity(activityType <- 'EventEndEmpty'),
        firstSequence : BPMN!SequenceEdge}

(3) rule Task{
     from a : BRM!Task
     to b : BPMN!Activity(activityType <- 'Task', name <- a.name)}

(4) rule ANDDecomposition_Sequence{ --Implementation of WCP-1
     from a : BRM!ANDDecomposition(self.type = #SequentialOrder)
     to b : BPMN!SequenceEdge(iD <- 'Sequence Edge')}

(5) rule ANDDecomposition_Parallel_FirstOccurrence{ --Implementation of WCP-2 and WCP-3
     from a : BRM!ANDDecomposition(self.type = #ParallelOrder and
                                   BRM!ANDDecomposition.allInstances()->first())
     to b : BPMN!Activity(activityType <- 'GatewayParallel'),
        c : BPMN!SequenceEdge(iD <- 'Left Parallel Edge'),
        d : BPMN!SequenceEdge(iD <- 'Right Parallel Edge'),
        e : BPMN!Activity(activityType <- 'GatewayParallel'),
        f : BPMN!SequenceEdge(iD <- 'Edge Closing Parallel Construction')}

(6) rule ANDDecomposition_Parallel_OtherOccurrences{ --Implementation of WCP-2 and WCP-3
     from a : BRM!ANDDecomposition(self.type = #ParallelOrder and not
                                   BRM!ANDDecomposition.allInstances()->first())
     to b : BPMN!SequenceEdge(iD <- 'Left Parallel Edge'),
        c : BPMN!SequenceEdge(iD <- 'Right Parallel Edge')}

(7) rule ORDecomposition_FirstOccurrence{ --Implementation of WCP-4 and WCP-5
     from a : BRM!ORDecomposition(BRM!ORDecomposition.allInstances()->first())
     to b : BPMN!Activity(activityType <- 'GatewayDataBasedExclusive'),
        c : BPMN!SequenceEdge(iD <- 'Left Conditional Edge'),
        d : BPMN!SequenceEdge(iD <- 'Right Conditional Edge'),
        e : BPMN!Activity(activityType <- 'GatewayDataBasedExclusive'),
        f : BPMN!SequenceEdge(iD <- 'Edge Closing Conditional Construction')}

(8) rule ORDecomposition_OtherOccurrences{ --Implementation of WCP-4 and WCP-5
     from a : BRM!ORDecomposition(not BRM!ORDecomposition.allInstances()->first())
     to b : BPMN!SequenceEdge(iD <- 'Left Conditional Edge'),
        c : BPMN!SequenceEdge(iD <- 'Right Conditional Edge')}
```