

BPM 2005

Third International Conference on Business Process Management

>>> *Nancy, France, 5-7 September 2005*

BPM 2005 is the third in a conference series that provides a forum for researchers and practitioners in all aspects of business process management.

Program Chairs

Ekkart Kindler
Markus Nüttgens

Program Committee

W. v. d. Aalst
J. Becker
J. v. Brocke
C. Bussler
P. Dadam
S. Dustdar
P. Loos
A. Oberweis
F. Rump
O. Thomas



First International Workshop on Business Process Reference Models (BPRM'05)

September 5, 2005

Pre-proceedings



Ekkart Kindler, Markus Nüttgens (eds.)

Business Process Reference Models

**Proceedings of the Workshop on
Business Process Reference Models (BPRM 2005)**

Satellite workshop of the
Third International Conference on Business Process Management (BPM)
Nancy, France, September 5, 2005

Volume Editors

Ekkart Kindler
Department of Computer Science
University of Paderborn
D-33100 Paderborn
Germany
email: kindler@upb.de

and

Markus Nüttgens
Universität Hamburg
WISO Fakultät
Wirtschaftsinformatik
Von-Melle-Park 9
D-20146 Hamburg
Germany
email: markus.nuettgens@wiso.uni-hamburg.de

Preface

Reference models for business processes have been a successful means for designing, redesigning, tailoring, and implementing business processes. Still there is no common understanding of reference models for business processes:

- What is a reference model?
- What makes them different from a business process model?
- What should be covered by a reference model?
- What is their purpose and how should they be used?
- How should they be designed and presented?

The workshop brings together people from different application areas, using different notations and formalisms, in order to present and discuss their point of view. The workshop should help

- to share experiences with the use of reference models,
- to better understand the purpose and the role of reference models,
- to identify the aspects that should be covered by reference models,
- to discuss notations and meta models for reference models, and, eventually,
- to come up with a technology to efficiently design and to use reference models.

We are happy that the workshop on Business Process Reference Models was accepted as a satellite event of BPM 2005, and we would like to thank the local organizers for their work and support in organizing this event. Moreover, we would like to thank the Program Committee and all referees for helping select and improve the contributions to this workshop. Many thanks also to all authors for their contributions.

July 2005, Ekkart Kindler and Markus Nüttgens

Program Committee

W. v. d. Aalst, Eindhoven, The Netherlands

J. Becker, Münster, Germany

J. v. Brocke, Münster, Germany

C. Bussler, Galway, Ireland

P. Dadam, Ulm, Germany

S. Dustdar, Vienna, Austria

E. Kindler (co-chair), Paderborn, Germany

P. Loos, Mainz, Germany

M. Nüttgens (co-chair), Hamburg, Germany

A. Oberweis, Karlsruhe, Germany

F. Rump, Emden, Germany

O. Thomas, Saarbrücken, Germany

Referees

Otmar Adam

Peter Fettke

Bettina Kaffai

Christian Seel, Dipl.-Kfm.

Christian Seel, MScIS

Jörg Zwicker

Table of Contents

| | |
|---|----|
| Business Process Reference Models: Survey and Classification | 1 |
| <i>P. Fettke, P. Loos, J. Zwicker</i> | |
| Understanding the Term Reference Model in Information Systems Research: History, Literature Analysis and Explanation | 16 |
| <i>O. Thomas</i> | |
| Towards a Reference Model for Work Distribution in Workflow Management Systems | 30 |
| <i>M. Pesic, W.M.P. van der Aalst</i> | |
| An Open and Formalism Independent Meta-Model for Business Processes | 45 |
| <i>B. Axenath, E. Kindler, V. Rubin</i> | |
| On the Syntax of Reference Model Configuration – Transforming the C-EPC into Lawful EPC Models | 60 |
| <i>J. Recker, M. Rosemann, W.M.P. van der Aalst, J. Mendling</i> | |
| Configurable Process Models as a Basis for Reference Modeling | 76 |
| <i>W.M.P. van der Aalst, A. Dreiling, F. Gottschalk, M. Rosemann, M.H. Jansen-Vullers</i> | |

Business Process Reference Models: Survey and Classification

Peter Fettke, Peter Loos, and Jörg Zwicker

Johannes Gutenberg University Mainz, Information Systems and Management,
55099 Mainz, Germany
{fettke, loos, zwicker}@isym.bwl.uni-mainz.de
<http://isym.bwl.uni-mainz.de>

Abstract. Within the Information Systems field, reference models have been known for many years. The aim of this paper is to survey and to describe well-known reference models for business processes. Our analysis of 30 process reference models is based on a framework consisting of criteria such as application domain, used process modeling languages, model's size, known evaluations and applications of process reference models. Furthermore, we identify model domains, which have been dealt with, describe similarities and differences between the available process reference models, and point to open research questions.

1 Introduction

Information modeling is a core vehicle to analyze, design, implement, and deploy information systems [1]. However, the modeling process is often resource consuming and faulty. As a way to cope with these failures and to improve the development of enterprise-specific models, the idea of reference modeling was born [2-4].

While an application model represents a particular enterprise system, a conceptual model represents a class of similar enterprise systems. It is a conceptual framework that can be used as a blueprint for information system construction [5]. To use a particular reference model, it must be adapted to the requirements of a particular enterprise. Reference models are also called universal models, generic models, or model patterns. The term reference model for business processes refers to a specific type of reference model. A process reference model represents dynamic aspects of an enterprise, e.g. activity sequences, organizational activities required to satisfy customer needs, control-flow between activities, particular dependency constraints etc. [6].

In publicly available sources, numerous more or less elaborated process reference models are proposed. The main objective of this paper is to identify, to survey and to describe the well-known process reference models. Compared to existing reference model surveys [5, 7], this study is more comprehensive and focuses on reference models for business processes.

Our study is of both practical and theoretical importance. From a practical point of view, the selection of an appropriate process reference model is difficult and complicated. One presumption of reusing a reference model is to know its availability, its

application domain, its potentials and limitations etc. A model survey can offer such information. Thus, this instrument fosters a rational and systematic model selection process.

Beside the practical relevance, surveys of reference models are of importance for the theory of enterprise modeling in general and for the theory of reference modeling in particular. Surveys of reference models can show varieties, gaps and areas of improvements. The results of a survey represent a meaningful basis for new and advanced reference models. Even if such an investigation does not take place in conjunction with the development of a new reference model, at least the scope of already developed reference models should be made clear by such a survey afterwards. Therefore, a survey of process reference model stimulates the scientific progress of reference modeling.

The paper unfolds as follows: Section 1 motivates this piece of research. We introduce a framework for describing and classifying process reference models in Section 2. In Section 3, we use this framework to describe 30 well-known process reference models. The obtained results are discussed in Section 4. The paper ends with some concluding remarks.

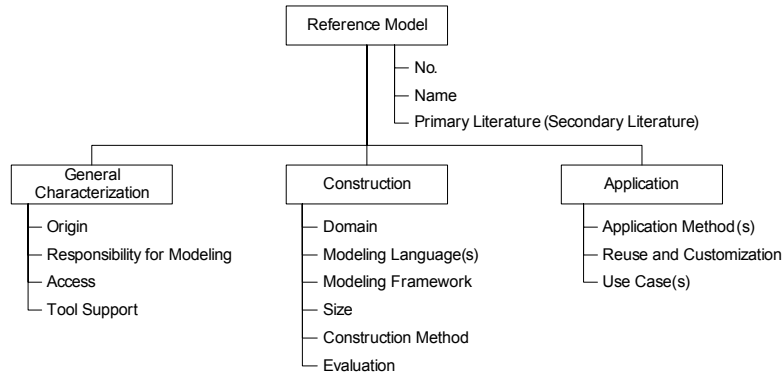


Fig. 1. Criteria for describing process reference models

2 Framework

Existing reference models could be structured regarding numerous points of view. Figure 1 illustrates and structures the here considered criteria for characterizing process reference models. Beside universal characteristics, suitable for the complete spread of reference models, the description and classification of process reference models requires particular consideration of process-related criteria. The several universal and process-specific criteria of the framework are described in the following:

- *Identification*: The identification of reference models is made by running numbers and reference model names. References, wherein the reference models are described, are also specified (primary literature). This information is completed with

additional references (secondary literature) wherein certain reference model properties are explained. The specification of secondary literature particularly supports to provide information about limited accessible reference models.

- *General Characterization*: To generally characterize a reference model, the origin, responsibility for modeling, access, and availability of tool support are stated.
 - *Origin*: The origin informs about the classification of the person(s) who have developed the reference model. In this regard, both science and practice can be distinguished.
 - *Responsibility for Modeling*: This criterion describes the persons or organizations developed the reference model.
 - *Access*: The access specifies the accessibility to the reference model by third parties. If the reference model is completely obtainable over usual ways of librarianship the access is classified as “open”. The access is “closed”, if the responsible person(s) or institution provides no possibility for using and recognizing the reference model by third parties. If the access is neither open nor closed the access is classified as “limited”. This is the case, e.g., if the reference model can be purchased as standalone product or it is accessible over an internet server, which does not belong to official librarianship. If the access to the reference model is closed the information of all aforementioned and following criteria is based on statements from the specified primary and secondary literature.
 - *Tool Support*: This criterion describes whether the reference model can be automatically used by a software tool or whether the reference model is only available in paper or digital copy.
- *Construction*: The following six criteria address the construction of process reference models:
 - *Domain*: The domain describes the field of application from perspective of the person(s) or institution responsible for developing the reference model. The criterion is distinguished into domain differentiation and domain description. Specifying the domain differentiation serves to distinguish varying principles of domain classification. So far, several differentiation approaches have been proposed. Using [8] in this framework, a widely elaborated approach is considered. With this, basically different principles of differentiation can be identified: Institutional differentiation is based on institutional characteristics of the intended business system (e.g. “Industrial Enterprise”, “Insurer” or “Bank”); functional differentiation is realized through business functions as differentiation characteristic (e.g. classical business functions like “Distribution Logistic”, “Production Planning and Control” or newer functions like “Facility Management”, “Knowledge Management”, “Controlling”); object-driven differentiation where business objects serves as differentiation characteristic (e.g. “Life Insurance” or “Branch Business”); enterprise type-driven differentiation is based on special enterprise characteristics (e.g. a book publisher can be considered as a special type of a publisher.). Also universal reference models exist which cannot be classified based on one of the aforementioned principles. Beside the domain differentiation, the domain description specifies the intended field of the reference model’s application using some words.
 - *Modeling Language(s)*: The language criterion states the modeling language(s) used to represent the reference model. To address the particular consideration

and description of process reference models, modeling languages or diagram types used to represent process models of the reference model are particularly specified. Further modeling languages are additionally described.

- *Modeling Framework*: This criterion describes whether a modeling framework is part of the reference model. A framework can structure relevant elements esp. diagrams of a reference model and their relationships at a higher level of abstraction. This serves the reduction of complexity and provides an overview of elements and relationships within the reference model.
 - *Size*: So far, appropriate size metrics for models of different modeling languages do not exist [7]. To give a vague impression about the size of the described reference models, several metrics can be used. The number of represented diagrams and views pose as general attributes. As a special process-related metric, the number of process steps within represented process diagrams is stated. The aforementioned sizes of smaller models (<30) are exactly counted, the size of bigger models are estimated. Estimated values have to be rounded off to full decade. If the access to the model is closed the information is based upon statements of given references.
 - *Construction Method*: This criterion states the modeling concept used by the responsible person(s) or institution for developing the reference model.
 - *Evaluation*: This criterion describes the used methods for evaluating the reference model by the person(s) or institution responsible for developing the reference model or by third parties. Evaluation methods are only considered, if they are explicitly intended for model evaluation by the evaluator. Besides the method, it is stated whether the result of performed evaluation is inter-subjective verifiable.
- *Application*: The following three criteria address the application of process reference models:
- *Application Method(s)*: This criterion describes the known method resp. concept for applying the reference model.
 - *Reuse and Customization*: This criterion lists concepts for reusing and customizing of model elements in the scope of the model's application.
 - *Use Case(s)*: The use case(s) describes how often the reference model was applied to construct an application model. Like to the evaluation method, the application of the reference model is also completed by the information whether the number and extent of use cases are inter-subjective verifiable.

3 Results

Reference models are represented in various modeling languages. From all reference models, process reference models are identified for this survey using the modeling language representing the reference models. If one of the reference models is represented in one or more established process modeling languages it is recorded as business process reference model. Within the survey, we identify and describe 30 well-known process reference models. These models are depicted in the table within the appendix of this paper.

3.1 General Characterization

In the following, the models are characterized regarding the criteria origin, responsibility for modeling, access and tool support:

- *Origin*: Practitioners developed eight reference models. The design of the other 22 models was partly or completely done by scientists.
- *Responsibility for Modeling*: Fifteen reference models were developed by one person, the author of the primary reference, and eight models by several persons. Furthermore, enterprises are responsible for the development of four models, two models were developed by associations and one model was created by an independent office of the British government.
- *Access*: The access is open to 16 reference models of the survey, closed to eight models and limited to six models.
- *Tool Support*: Fifteen reference models are only published as paper copy, and eight models can directly be handled in a tool. Finally, no statement is given by the responsible person(s) to the remaining seven models.

3.2 Construction

Domain. Due to missing standards, reference models cannot be described based on a consistent domain framework. A classifying description of the results and an associated quantitative analysis is only possible through the subjective-driven specification of the domain differentiation. So, 11 reference models are provided for institutional context. Further 12 models are classified into the functional context. Finally, seven models cannot be exactly classified based on the differentiation principle as described in chapter 2.

Using the domain description from the responsible designer(s), reference models for the information systems' development in industrial enterprises exist (e.g. "Aachener PPS"-model or SCOR). Further subjects of reference modeling are financial service providers (e.g. insurer or banks), book publishers or special business functions like knowledge management, logistic and environmental data management.

Modeling Language(s). To represent reference models, several modeling languages are used. Widely-accepted modeling languages, such as the Entity-relationship Model (ERM) and the Unified Modeling Language (UML), are applied. Furthermore, modeling languages like the Semantic Object Model (SOM), function trees or special object-oriented languages, are used. Some designers use modeling languages which are exclusively developed to construct the correspondent reference model.

To model the business process view of the reference models, Event-driven Process Chains (EPC) are used in many cases. Also parts of further languages and language frameworks are utilized to design the processes. For example, the activity and use case diagrams as parts of the UML are particularly often used. SOM and the Multi-Perspective Enterprise Modeling (MEMO) similarly possess views for business process modeling. Further languages and diagram types are e.g. the process chain diagram (VKD), value chain diagram, task chain diagram or proprietary languages.

Modeling Framework. To structure the elements and relationships of the process reference models, modeling frameworks can be used. In 18 cases, the designer(s) of the models and/or the authors of the literature references make statements about a framework as part of the reference model. In 10 of the 18 cases, the author explicitly negates the existence of an appropriate framework. In the remaining eight cases, the existence of a framework is stated or it is represented and described.

Size. While determining of model's size, values have only been acquired, if the process reference models are represented within the given references or the author(s) specifies the appropriate number. Where the metrics could be determined: The number of used diagrams ranges from one up to estimated 450, whereas in the most cases the number does not exceed 50 diagrams. The number of views ranges from one to four. Finally, the number of process steps as process-related size ranges from estimated 50 to 300 and in one case to 1500 steps.

Construction Method. Statements on the development process are identified regarding 14 of the analyzed process reference models. Four of these cases explicitly refer to a used procedure model, build up on such a model or introduce an own model. The designers of the remaining 10 reference models circumscribe the applied procedure using only few words without comprehensively explicit the chosen procedure. For example, the designers describe their reference models as "deductive derived" or "constructed within the scope of case studies" resp. "constructed on case examples".

Evaluation. Methods and procedures for evaluating the quality are only determined for 15 reference models. These cases can be distinguished into two groups:

1. *Evaluation approach:* Information on how an evaluation of the reference models can be done is stated in two cases. The information is depicted without documenting concrete results. The proposals cover a comparison of a reference model with an enterprise or the annotation of a necessary empirical evaluation.
2. *Results:* In regard to 13 reference models, results of evaluations are described by authors of according literature references. The results base on different procedures:
 - In three cases, the reference model was evaluated through the prototypical implementation within a software product.
 - In three cases, the reference model was used for case studies in more or less real conditions: The spread ranges from simple, fictitious examples to reality-similar utilizations.
 - In one case, a questioning of model users was organized to determine the possibilities of utilizations.
 - In two cases, an ad hoc evaluation was carried out compiling several arguments to show preferences and limitations of the reference model from the view of the evaluator.
 - In two cases, a thought experiment was performed by the author. This is a way to evaluate the reference model through demonstrating exemplary application within a hypothetical context.
 - In further two cases, a prototypical or exemplary application of the reference model within a fictitious context is described without a real application.

3.3 Application

Application Method. Possible potentials of process reference modeling only unfold with applying the reference models. Thirteen of the identified process reference models cover proposals including configurational options for the model's application process. Most of them (twelve authors) develop a model-based procedure model for specific application purposes. Typical examples are reference model-based procedure model: for knowledge management [9], for developing information and communication architecture [10]. For the SAP R/3 reference model, a model-based procedure model is not proposed. Instead, contributions exist wherein the application of the model is exemplary described [11-13].

Reuse and Customization. Statements on concepts for reusing and customizing of elements within the reference models are only provided with nine of the entire 30 models. Similar to the modeling language, one reference model can comprise more than one concept. The specialization of the developed models and the usage of build-time operators are often used concepts. Beside others, a particular case is the usage of model variants for different application contexts in one model.

Use Case(s). Use cases are also a way of evaluating; similar to case studies but independently realized. The real application projects are not construed as ex ante evaluating studies rather than the project results are used as ex post evaluation. In nine of the entire 30 cases, the reference models were used within real projects. In the remaining 21 cases, statements on real applications are not available, although, in one case, the author explicitly states that no real application has taken place.

4 Discussion

4.1 Identified Reference Models

Within this survey, 30 process reference models are described and classified. The quantification does not raise the claim of a comprehensive survey. In fact, because of the lack of space, this paper shall document 30 well-known process reference models. From a practical point of view, the determination whether a reference model is a process reference model, is difficult and bases on subjective distortions:

- It is complicated to answer whether a reference model is a collection of many individual models or an overall model. For example, it can be argued that the SKO-reference model consists of two reference models, the SKO-reference *data* model and the SKO-reference *process* model. Moreover, it is unclear how to describe model variants; either as part of a comprehensive reference model or as several individual reference models.
- Further difficulties arise with presenting one reference model in several publications. A decision is necessary whether these models are similar representations of different reference models or different representations resp. versions of the same

model. Moreover, interpretational problems arise with incompletely published models.

- Finally, the decision whether a “reference model” is a reference model in the here implied intuitive conception, is often complicated.

4.2 General Characterization

In publicly available sources, numerous more or less elaborated process reference models are proposed. Most of them were developed within science. In spite of this, it can be suspected that process reference models can be found in the reality of enterprise modeling. Nevertheless, the survey and classification illustrates a lack of implementation of the analyzed reference models in real environments. This can depend on several reasons:

- Many reference models are partly not accessible or only limited accessible. This fact is plausible in regard to reference models developed within practice, but the limitation of reference models from science is inappropriate. Moreover, 20 of the analyzed reference models with origin in science do not possess tool support. From the view of reference modeling objectives, the propagation and application of the models is prohibited by the lack of access and missing tool support. In fact, it is necessary to fully publish the models for practical application and provide adequate tool support.
- From a practical point of view, the selection of an appropriate process reference model is difficult and complicated. One presumption of reusing a reference model is to know its availability and application-relevant information. It can be suspected that many available reference models, in particular the models developed in science, are not public. Application methods resp. procedure models to apply reference modeling in practical context do not regard the multiplicity of existing models; at least we do not know such methods resp. procedures. Moreover, the field of reference modeling is not comprehensively established within practical enterprise modeling.

Further reasons for the low application of the reference models are associated with characteristics like used representation language, possibility of inter-subjective evaluation, existence of appropriate application method etc. Primary gaps are identified within the correspondent, following sections.

4.3 Construction

Domain. The differentiation of application domains is done in several ways. Using the principles of differentiation introduced in chapter 2, the analyzed reference models can only be classified as institutional and functional domains. Furthermore, some reference models cannot be exactly classified. These two issues do not show a lack of reference models for certain domains. In fact, it points out difficulties regarding the differentiation. So, the introduced differentiation criteria do not exclude from each other but partly overlap. For example, a book publisher can be both a special enterprise type and an institution. Also enterprises which perform production planning and

control functions are regularly classified as industry. The difficulty cannot be deepened any further. Instead, it shall be stressed that the differentiation of a reference model is not trivial and it has to be done with utmost diligence: Finally, the differentiation of the domain determines the intended field of application and as consequence the reference model's potential.

Modeling Language(s). Reasons like more or less objective properties, personal preferences, available tools etc. hamstring the development of standardized modeling languages. Nevertheless, it is desirable to explicit the special requirements to a modeling language before using to design a reference model. Only this guarantees inter-subjective proving and makes the language a subject of criticism. It should also be pointed out which constructs a language has to provide for serving as efficient reference modeling language. Furthermore, the question of whether configuration mechanism for modeling languages can be usefully constituted in reference modeling exists. Currently, only few authors evaluate languages before designing a reference model.

Construction Method. So far, only few authors explicit the procedure of their model's construction. Two types of procedure models can be generally distinguished:

- Empiric-oriented design methods develop reference models based on a class of real enterprises.
- Deductive-oriented design methods derive a reference model from formal-logical and mathematical inferences.

A rating of both procedures is ambivalent: Empiric-oriented methods neglect possible, but, up to now, unrealized design concepts of business systems. On the other hand, deductive methods suggest a compelling nature which is not present within the reality of model design. Although, this procedure does not perform invulnerable inferences, it based on simple plausibility deliberations.

Advantages and disadvantages of both procedures shall not be discussed in more detail. Though, we do not know any work investigating this problem from an empirical point of view. An intensive analysis of effects of several design methods for reference model is desirable.

Evaluation. The evaluation of reference models is of high importance and an extraordinary challenge. Both acceptable evaluation criteria and methods are not established [14]. On the one hand, the scientific perspective demands precise, consistent and complete reference models. On the other hand, from perspective of application, simplicity and understandability are of relevance. Hence, conflicts of objectives can arise.

Although, several results of reference model evaluation exist, considerable more need for research is noticed. Existing evaluation results cannot often be evaluated by third parties. For example, some evaluators only argue that the reference model stands the practical application. Furthermore, standardized methods and criteria for evaluation do not exist.

Meanwhile upcoming contributions with reference model evaluation by third parties and by the author are a favorable development (e.g. [15]). In some cases, these contributions can be critical assessed from a methodical point of view (e.g. low validity), but we recommend such evaluations: These contributions are essential to evaluate the potentials of reference models. Only evaluations by third parties ensure the reference models' independency and usability.

The evaluation by third parties is often failed because of practical limitations: As already mentioned and criticized in section 4.2, the reference models are partly not accessible or only limited accessible. For evaluating the reference models by third parties, it is necessary to fully publish the models.

4.4 Application

As several procedures are known within the field of reference model design, no standardized application methods have been presented. Although, it is obvious that only one design method can be used for the development of one reference model, it is possible that several application methods can be used to apply a reference model. Nevertheless, the realization of similar task at the reference models' application can be assumed:

- *Selection of one reference model*: Identified application methods abstract from this question and only take one given reference models into account. Although, Schwegmann originally proposes to decide according to instinct whether a new reference model shall be developed or an existing reference model shall be selected and reused [16].
- *Configuration and adaptation of one reference model*: For this purpose, approaches like a configurational reference modeling are described [17], although they are not widely implemented by existing reference models.

5 Conclusion and Further Research

This paper analyzes the body of process reference models available in public sources. The main contribution of our work is three-folded: First, we propose a new framework to describe business process reference models. In this study, we use this framework to survey well-known business process reference models. However, the proposed framework can be used to guide the developing process of *new* models, too. Second, we demonstrate the applicability of the framework by describing 30 business process reference models. This survey fosters the model selection process during application model development. Third, our analysis of the obtained results points to open research questions. For instance, the development of language constructs for reusing and customizing of model elements in the scope of the model's application or the evaluation of languages before designing a reference model.

Our work has some limitations: First, we do not introduce the term reference model formally. Instead, our analysis is based on a rather intuitive conception, which may lead to misunderstandings. However, we believe it is not easy to give an acceptable explication of the term reference model, because, e.g., the term is both used as a one- and two-place predicate [18]. Second, our survey is mainly based on a literature review. We suspect that business process reference models can be found in the reality of enterprise modeling, too. However, we only survey models that are already described in literature. So, our study is just based on secondary information. Third, the framework used to describe reference models is limited. May be, it will be necessary to

extend the framework and to define the used criteria in a more rigorously way. Also the justification of the used criteria has to be more stringent in future.

In the future, we try to overcome the mentioned limitations. Our long-term research objective is to develop the conceptual foundations for reference model catalogs. Reference model catalogs are inspired by construction catalogs used in engineering disciplines and provide systematic and comprehensive information about all known reference models. To achieve this objective, we will develop a formalized notion of the term reference model first. Second, we are preparing empirical studies to describe and to explain reference modeling processes found in reality. Third, we will use ontology technology to capture our framework and to describe the known body of reference models.

Acknowledgement. This paper presents results from the research project “Reference modeling with reference model catalogs” funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation).

References

1. Wand, Y., Weber, R.: Research Commentary: Information Systems and Conceptual Modeling - A Research Agenda. *Information Systems Research* 13 (2002) 4, 363-377
2. Mertins, K., Bernus, P.: Reference Models. In: Schmidt, G. (ed.) *Handbook on Architectures of Information Systems*. Springer, Berlin et al. (1998) 615-617
3. Mišić, V. B., Zhao, J. L.: Evaluating the Quality of Reference Models. In: Storey, V. C. (ed.) *Conceptual Modeling - ER 2000 - 19th International Conference on Conceptual Modeling*, Salt Lake City, Utah, USA, October 9-12, 2000 Proceedings. Springer, Berlin et al. (2000) 484-498
4. Scheer, A.-W., Nüttgens, M.: ARIS Architecture and Reference Models for Business Process Management. In: Oberweis, A. (ed.) *Business Process Management - Models, Techniques, and Empirical Studies*. Springer, Berlin et al. (2000) 376-389
5. Fettke, P., Loos, P.: Classification of reference models - a methodology and its application. *Information Systems and e-Business Management* 1 (2003) 1, 35-53
6. Becker, J., Kugeler, M., Rosemann, M.: *Process Management*. Springer (2003)
7. Van Belle, J.-P. W. G. D.: *A Framework for the Analysis and Evaluation of Enterprise Models*. University of Cape Town (2003)
8. Mertens, P., Lohmann, M.: Branche oder Betriebstyp als Klassifikationskriterien für die Standardsoftware der Zukunft? Erste Überlegungen, wie künftig betriebswirtschaftliche Standardsoftware entstehen könnte. In: *Verbundtagung Wirtschaftsinformatik 2000*, 110-135
9. Warnecke, G., Gissler, A., Stammwitz, G.: Referenzmodell Wissensmanagement - Ein Ansatz zur modellbasierten Gestaltung wissensorientierter Prozesse. *IM Information Management & Consulting* 13 (1998) 1, 24-29
10. Rohloff, M.: Das Prozessrahmenwerk der Siemens AG: Ein Referenzmodell für betriebliche Geschäftsprozesse als Grundlage einer systematischen Bebauung der IuK-Landschaft. In: Becker, J., Knackstedt, R. (eds.): *Wissensmanagement mit Referenzmodellen: Konzepte für die Anwendungssystem- und Organisationsgestaltung*. Physica-Verlag, Heidelberg (2002) 227-235
11. Keller, G., Lietschulte, A., Curran, T. A.: Business Engineering mit den R/3-Referenzmodellen. In: Nüttgens, M. (ed.) *Electronic Business Engineering - 4. Internationale Tagung Wirtschaftsinformatik 1999*. Physica-Verlag, Heidelberg (1999) 397-423

12. Lietschulte, A., Keller, G.: Modellgestützte R/3 Einführung. In: Mertens, P. (ed.) Referenzmodellierung '98: Anwendungsfelder in Theorie und Praxis, 14. Juli 1998. Forschungsinstitut für Rationalisierung an der RWTH Aachen, Aachen (1998) 5-1 - 5-8
13. Curran, T. A., Keller, G.: SAP R/3 Business Blueprint - Business Engineering mit den R/3-Referenzprozessen. Addison-Wesley, Bonn et al. (1999)
14. Fettke, P., Loos, P.: Multiperspective Evaluation of Reference Models - Towards a Framework. In: Jeusfeld, M. A., Pastor, Ó. (eds.): Conceptual Modeling for Novel Application Domains - ER 2003 Workshops ECOMO, IWCMQ, AOIS, and XSDM, Chicago, IL, USA, October 13, 2003. Springer, Berlin et al. (2003) 80-91
15. Taylor, C., Probst, C.: Business Process Reference Model Languages: Experiences from BPI Projects. In: Proc. INFORMATIK 2003 - Innovative Informatikanwendungen, Band 1, Beiträge der 33. Jahrestagung der Gesellschaft für Informatik e.V. (GI), 29. September - 2. Oktober 2003 in Frankfurt am Main (2003) 259-263
16. Schwegmann, A.: Objektorientierte Referenzmodellierung - Theoretische Grundlagen und praktische Anwendung. DUV, Wiesbaden (1999)
17. Becker, J., Delfmann, P., Dreiling, A., Knackstedt, R., Kuropka, D.: Configurative Process Modeling - Outlining an Approach to Increased Business Process Model Usability. In: Proc. Information Resources Management Association Conference (2003) 615-619
18. Fettke, P., Loos, P.: Referenzmodellierungsforschung. Wirtschaftsinformatik 46 (2004) 5, 331-340
19. Luczak, H., Kees, A.: Das Aachener PPS-Modell. In: Proc. Referenzmodellierung '98: Anwendungsfelder in Theorie und Praxis, 14. Juli 1998, RWTH Aachen (1998) 2-1 - 2-9
20. ten Voorde, H.: Dynamic Enterprise Modeling. In: Proc. Referenzmodellierung '98 - Anwendungsfelder in Theorie und Praxis, 14. Juli 1998, RWTH Aachen (1998) 7-1 - 7-22
21. Krcmar, H., Dold, G., Fischer, H., Strobel, M., Seifert, E. K.: Informationssysteme für das Umweltmanagement - Das Referenzmodell ECO-Integral. Oldenbourg (2000)
22. Frank, U.: Modeling Products for Versatile E-commerce Platforms - Essential Requirements and Generic Design Alternatives. In: Arisawa, H., Kambayashi, Y., Kumar, V., Mayr, H. C., Hunt, I. (eds.): ER 2001 Workshops, HUMACS, DASWIS, ECOMO, and DAMA. Springer, Berlin et al. (2002) 444-456
23. Becker, J., Schütte, R.: Handelsinformationssysteme. Landsberg/Lech (1996)
24. Hochstein, A., Hunziker, A.: Serviceorientierte Referenzmodelle des IT-Managements. HMD - Praxis der Wirtschaftsinformatik (2003) 232, 45-56
25. Kaiser, T. M., Bach, V., Vogler, P., Österle, H.: Eine Methode für die Konzeption von Intranets. HMD - Praxis der Wirtschaftsinformatik (1999) 209, 94-104
26. Buchwalter, J.: Elektronische Ausschreibungen in der Beschaffung - Referenzprozeßmodell und prototypische Realisierung. Eul, Lohmar, Köln (2002)
27. Gerber, S., Mai, A.: Ein Referenzmodell für das Filialgeschäft von Banken als betriebliche Wissensplattform. In: Becker, J., Knackstedt, R. (eds.): Wissensmanagement mit Referenzmodellen: Konzepte für die Anwendungssystem- und Organisationsgestaltung. Physica-Verlag, Heidelberg (2002) 195-206
28. Haas, C., Ahlemann, F., Hoppe, U.: Organisationale Integration von E-Learning in Unternehmen - ein Referenz-Informationsmodell. In: Schoop, E. (ed.) Wirtschaftsinformatik 2003/Band I - Medien - Märkte - Mobilität. Physica, Heidelberg (2003) 707-726
29. Herrmann, G.: Verlässlichkeit von Geschäftsprozessen - Konzeptionelle Modellbildung und Realisierungsrahmen. Logos, Berlin (2002)
30. Kluger, M. A.: Beitrag zur effizienten Anwendung der dynamischen Unternehmensmodellierung. Verlag Praxiswissen, Dortmund (1999)
31. Krömker, M.: Werkzeug zur durchgängigen Systemunterstützung der Angebotserstellung in der Unikat- und Kleinserienfertigung. Verlag Mainz, Aachen (2000)
32. Kruse, C.: Referenzmodellgestütztes Geschäftsprozessmanagement: ein Ansatz zur prozessorientierten Gestaltung vertriebslogistischer Systeme. Gabler, Wiesbaden (1996)

33. Mertens, P.: Integrierte Informationsverarbeitung 1 - Administrations- und Dispositionssysteme in der Industrie. 12 edn. Gabler, Wiesbaden (2000)
34. Mertens, P., Gries, J.: Integrierte Informationsverarbeitung 2 - Planungs- und Kontrollsysteme in der Industrie. 9 edn. Gabler, Wiesbaden (2002)
35. Neumann, S.: Workflow-Anwendungen in technischen Dienstleistungen - Eine Referenz-Architektur für die Koordination von Prozessen im Gebäude- und Anlagenmanagement. Logos, Berlin (2003)
36. Pumpe, D.: Ein Referenzmodell zur Planung und Steuerung der Abläufe in Seehafen-Containerterminals. Mensch-und-Buch, Berlin (2000)
37. Remme, M.: Konstruktion von Geschäftsprozessen: ein modellgestützter Ansatz durch Montage generischer Prozesspartikel. Gabler, Wiesbaden (1997)
38. Rüffer, T.: Referenzgeschäftsprozeßmodellierung eines Lebensversicherungsunternehmens. In: Proc. Modellierung betrieblicher Informationssysteme - Proceedings der MobIS-Fachtagung 1999, 14. und 15. Oktober 1999, Universität Bamberg (1999) 86-107
39. Schaich, C.: Informationsmodell zur fachübergreifenden Beschreibung intelligenter Produktionsmaschinen. Utz, München (2000)
40. Schlagheck, B.: Objektorientierte Referenzmodelle für das Prozess- und Projektcontrolling - Grundlagen - Konstruktion - Anwendungsmöglichkeiten. DUV, Wiesbaden (2000)
41. Tzouvaras, A.: Referenzmodellierung für Buchverlage: Prozess- und Klassenmodelle für den Leistungsprozess. Cuvillier, Göttingen (2003)
42. Keller, G., Teufel, T.: SAP R/3 Process Oriented Implementation - Iterative Process Prototyping. Addison-Wesley, Harlow et al. (1998)
43. Gerber, S., Hiestermann, A., Kittlaus, H.-B.: Management von Prozeßmodellen dezentraler BPR-Projekte mit Hilfe eines zentralen Referenzprozeßmodells. In: Nüttgens, M. (ed.) Electronic Business Engineering - 4. Internationale Tagung Wirtschaftsinformatik 1999. Physica, Heidelberg (1999) 375-395
44. Eisenreich, A.: Das SKO-Datenmodell - ein Referenzmodell für die Sparkassenorganisation. In: Becker, J., Knackstedt, R. (eds.): Referenzmodellierung 2002: Methoden - Modelle - Erfahrungen. Institut für Wirtschaftsinformatik der Westfälischen Wilhelms-Universität Münster, Münster (2002) 121-132
45. Gerber, S., Müller-Luschnat, G.: Sind Referenzprozeßmodelle in der betrieblichen Praxis sinnvoll? - Ein Beispiel aus der Dienstleistungsbranche. In: Schürr, A. (ed.) Modellierung '99 - Workshop der Gesellschaft für Informatik e. V. (GI), März 1999 in Karlsruhe. Teubner, Stuttgart, Leipzig (1999) 27-42
46. Supply-Chain Council Inc. SCOR Overview. Overview of the SCOR Model v3.0.[Online]. Available: www.supply-chain.org
47. Stephens, S.: The Supply Chain Council and the Supply Chain Operations Reference Model. Supply Chain Management 1 (2001) 1, 9-13
48. Holten, R., Melchert, F.: Das Supply Chain Operations Reference (SCOR)-Modell. In: Becker, J., Knackstedt, R. (eds.): Wissensmanagement mit Referenzmodellen: Konzepte für die Anwendungssystem- und Organisationsgestaltung. Physica-Verlag, Heidelberg (2002) 207-226
49. Huan, S. H., Sheoran, S. K., Wang, G.: A review and analysis of supply chain operations reference (SCOR) model. Supply Chain Management - An International Journal 9 (2004) 1, 23-29
50. GDV (Ed.). Anwendungsarchitektur der deutschen Versicherungswirtschaft. Gesamtverband der deutschen Versicherungswirtschaft e. V. [Online]. Available: <http://www.gdv-online.de/vaa/>
51. Scheer, A.-W.: Wirtschaftsinformatik - Referenzmodelle für industrielle Geschäftsprozesse. 7 edn. Springer, Berlin et al. (1997)
52. Scheer, A.-W.: 20 Jahre Gestaltung industrieller Geschäftsprozesse. Industrie Management 20 (2004) 1, 11-18

Appendix

| Identification | | | General Characterization | | | | Construction | | |
|----------------|---|--|--------------------------|---|---------|--------------|------------------------|---|--|
| No. | Name | Primary Literature (Secondary Literature) | Origin | Responsibility for Modeling | Access | Tool Support | Domain Differentiation | Domain Description | Modeling Language(s) |
| 1 | "Aachener PPS"-Model | [19] | Science | Authors | Closed | Yes | Function | Production, Planning and Control Systems | Proprietary Process Model |
| 2 | Baan Reference Model | [20] | Practice | Baan | Closed | Yes | Others | n.S. | Proprietary Process Model |
| 3 | ECO-Integral | [21] | Science | Authors | Open | No | Function | Operational Environmental Protection | EPC |
| 4 | Enterprise Modeling for E-Commerce (ECOMOD) Reference Model | [22] | Science | Authors | Limited | n.S. | Others | Internet Platform for Commerce | MEMO-OrgML |
| 5 | "Handels-IT"-Model | [23] | Science | Authors | Open | No | Institution | Enterprises doing Commercial Functions | EPC |
| 6 | Information Technology Infrastructure Library (ITIL) | [(15, 24)] | Practice | Office of Government Commerce | Limited | No | Function | IT-Management | Verbal |
| 7 | PROMET-INET Reference Model | [25] | Science | Authors | Closed | No | Others | Intranet Conception | Proprietary Process Model |
| 8 | Process Framework of Siemens AG | [10] | Practice | Siemens AG | Closed | n.S. | Others | Development of Information and Communication Landscape | Graphical and Verbal |
| 9 | Buchwalter's Reference Model | [26] | Science | Author | Open | No | Function | Electronical ITB-Systems in Procurement | Value Chain Diagram, Task Chain Diagram |
| 10 | Reference Model of Gerber/Mai | [27] | Practice | Authors | Closed | Yes | Institution | Branch Business of Banks | Process Hierarchy-Diagrams |
| 11 | Reference Model of Haas et al. | [28] | Science | Authors | Closed | n.S. | Function | E-Learning Processes in Enterprises | EPC |
| 12 | Herrmann's Reference Model | [29] | Science | Author | Open | n.S. | Others | Reliability Requirements for Business Processes | UML Activity Diagram |
| 13 | Kluger's Reference Model | [30] | Science | Author | Limited | Yes | Function | Vehicle-based Transport System | Proprietary Process Model |
| 14 | Krömer's Reference Model | [31] | Science | Author | Open | n.S. | Institution | Creation of Offers for Unicorns and Small-sized Series | IDEF0 with Process Character |
| 15 | Kruse's Reference Model | [32] | Science | Author | Open | No | Function | Distribution Logistic | EPC |
| 16 | Reference Model of Mertens/Giese | [33, 34] | Science | Author | Open | No | Institution | Industrial Enterprise | EPC |
| 17 | Neumann's Reference Model | [35] | Science | Author | Open | No | Function | Technical Facility Management | EPC |
| 18 | Pumpe's Reference Model | [36] | Science | Author | Open | No | Institution | Seaport Container Terminal | EPC |
| 19 | Remme's Reference Model | [37] | Science | Author | Open | No | Others | Management Organization | EPC |
| 20 | Rüfner's Reference Model | [38] | Science | Author | Open | No | Institution | Semantic Object Model (SOM) using Interaction-Schema (IAS) for Structure and Transaction-Event-Schema (VES) for Dynamic | |
| 21 | Schlich's Reference Model | [39] | Science | Author | Open | n.S. | Institution | Primary Insurer at the Example of Life Insurance Domain | UML Use Case Diagram |
| 22 | Schlagheck's Reference Model | [40] | Science | Author | Open | No | Function | Production Machinery | UML Activity Diagram |
| 23 | Schwegmann's Reference Model | [16] | Science | Author | Open | No | Function | Controlling | UML Activity Diagram |
| 24 | Tzouvaras's Reference Model | [41] | Science | Author | Open | No | Institution | Warehouse Management | EPC |
| 25 | Reference Model of Warnecke et al. | [9] | Science | Authors | Closed | n.S. | Function | Service Processes at Book Publishers | UML Activity Diagram |
| 26 | SAP R/3 Reference Model | ARIS for R/3 of IDS Scheer AG [42] [(11-13)] | Practice | SAP AG | Limited | Yes | Others | Knowledge Management | Proprietary Process Model |
| 27 | "Sparkassenorganisation (SKO)"-Reference Model | [43-45] | Practice | Information Center of "Sparkassen-organisation GmbH" | Closed | Yes | Institution | n.S. | EPC |
| 28 | Supply Chain Operations Reference Model (SCOR-Model) | [46] [(47-49)] | Practice | Supply Chain Council Inc. | Limited | Yes | Function | German "Sparkassen" | EPC |
| 29 | Insurance Architecture (VAA) | [50] | Practice | Gesamtverband der deutschen Versicherungswirtschaft e. V. (GDV, German Insurance Association) | Limited | Yes | Institution | Supply Chain Management | Graphical and Verbal |
| 30 | Y-CIM Model | [51] [(52)] | Science | Author | Open | No | Institution | Insurer | Verbal Description, UML Use Case Diagram |

Legend:
 * - Number is estimated
 n.S. - no statement

| No. | Modeling Lang. | Modeling Framework | Construction Size | | | Construction Method | Evaluation / Inter-subjective Verifiable | Application | | |
|-----|--|--------------------|--------------------|-----------------|----------------------|---|--|--|---|--|
| | Further Language(s) | | Number of Diagrams | Number of Views | Process-related Size | | | Application Method(s) | Reuse and Customization | Use Case(s)/ Inter-subjective Verifiable |
| 1 | Task Model, Function Model, Data Model, Object Model | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | Multiple / No |
| 2 | Funktion Model, Organizational Model | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. |
| 3 | Function Tree, ERM | Yes | 100* | 3 | 230* | Case studies | Case Studies / No | Procedure Model | n.S. | 3 / No |
| 4 | MEMO | n.S. | n.S. | n.S. | n.S. | n.S. | Prototype, Critical Argumentation / Partly | n.S. | n.S. | n.S. |
| 5 | ERM, Function Tree | Yes | 100* | 3 | 1500* | n.S. | n.S. | Procedure Model for Development of an Information Strategy | Variants | n.S. |
| 6 | Verbal | Yes | n.S. | n.S. | n.S. | n.S. | Questioning [15] / Yes | n.S. | n.S. | According to [15] Multiple Applied / No |
| 7 | | n.S. | n.S. | n.S. | n.S. | n.S. | Case Studies / Partly | PROMET-related Procedure Model | n.S. | n.S. |
| 8 | | Yes | n.S. | n.S. | n.S. | n.S. | n.S. | Procedure Model for Developing of Information and Communication Architecture | n.S. | Real Application / No |
| 9 | | No | 16 | 2 | 130 | Analysis of Existing Systems | Prototype / Partly | n.S. | n.S. | n.S. |
| 10 | Class Diagrams | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. |
| 11 | ERM | n.S. | n.S. | n.S. | n.S. | Case Examples | Empirical Verification is proposed | n.S. | n.S. | n.S. |
| 12 | UML | No | n.S. | n.S. | n.S. | Schütte's Procedure Model | n.S. | n.S. | n.S. | 0 / No |
| 13 | Data Model | n.S. | n.S. | n.S. | n.S. | Following the Construction Method for Technical Products (VDI 2222) | Prototypical Application | Procedure Model | n.S. | 1 / Partly |
| 14 | | No | 16 | 1 | - | Actual Survey and Weak-point Analysis | n.S. | Procedure Model for Introducing | n.S. | 3 / Yes |
| 15 | Function Tree, ERM, Organigram | Yes | 12 | 4 | 70 | n.S. | n.S. | Procedure Model | Composition of Reference Modules, Customization of Model Contents | n.S. |
| 16 | Function Tree, ERM | No | 1 | 1 | - | n.S. | n.S. | n.S. | n.S. | n.S. |
| 17 | Value Chain, ERM | No | 50 | 3 | 210* | Analysis of Existing Reference Models | Thought Experiment / Yes | n.S. | Process Extensions | n.S. |
| 18 | Class Diagrams | No | 19 | 2 | 50* | Empirical | Ad Hoc Evaluation / Partly | n.S. | n.S. | n.S. |
| 19 | | No | 9 | 1 | 50* | Analysis of Design Decisions | Thought Experiment / Partly | Procedure Model | Placeholder and Specialization | n.S. |
| 20 | | No | 8 | 3 | - | Deductive | Model Comparison in Practice (Proposal) | n.S. | n.S. | n.S. |
| 21 | UML | n.S. | n.S. | n.S. | n.S. | Balzer's Object-oriented Analysis (OOA) | Exemplary Application | n.S. | n.S. | n.S. |
| 22 | UML Class Diagram | Yes | 20 | 2 | - | Procedure Model | Prototype / Partly | Procedure Model | Model Specialization, Build-time Operators | n.S. |
| 23 | UML Class Diagram | No | 16 | 2 | 80* | Procedure Model | Ad Hoc Evaluation / Partly | Procedure Model | Model Specialization, Build-time Operators | n.S. |
| 24 | UML, Value Chain Diagram | Yes | 32 | 2 | - | Procedure Model | Two Case Studies / Partly | n.S. | Build-time Operators | n.S. |
| 25 | Object Model | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | Procedure Model | n.S. | n.S. |
| 26 | ERM, Function Tree | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | [11-13] | n.S. | n.S. |
| 27 | Function Tree, ERM | Yes | n.S. | n.S. | n.S. | n.S. | n.S. | Procedure Model | Modeling Level, Specialization | According to [44] 30 / No |
| 28 | | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | Specialization | Multiple |
| 29 | ERM, Function Tree, UML Class Diagram | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. | n.S. |
| 30 | ERM, Function Tree | Yes | 450* | 4 | 300* | n.S. | n.S. | n.S. | n.S. | According to [52] Multiple / No |

Legend:
 * - Number is estimated
 n.S. - no statement

Understanding the Term Reference Model in Information Systems Research: History, Literature Analysis and Explanation

Oliver Thomas

Institute for Information Systems (IWi)
at the German Research Center for Artificial Intelligence (DFKI),
Saarbruecken (Germany)
thomas@iwi.uni-sb.de
<http://www.iwi.uni-sb.de>

Abstract. The heart of every scientific discipline is its own unique, uniform and acknowledged terminology. As an application-oriented mediator between business administration and computer science, information systems research in particular is in need of a theoretical foundation and an instrument capable of translating basic theoretical knowledge into practical applications. Its dependency on and proximity to actual practice, as well as the rapid development of information technology often get in the way of the sound, systematic and consistent formation of concepts. Reference modeling is especially in need of a theoretical foundation. Due to the strong influence of implementation-oriented thought within this field, a gap has resulted between research and practice which has often led to undesirable developments. The high expectations organization and application system developers have on the reutilization of reference models are often disappointed. Apparently, the recommendations made by reference model developers often do not meet the expectations of potential model-users. One reason for this is the non-uniform grasp of the term reference model. This article attempts to counteract this deficiency by way of a detailed analysis of the way the term reference model is used and understood.

1 Initial Situation and Problem

Information systems are mediators between business frameworks and information technology and can be characterized using in-depth system-theoretical attributes. For example, the complexity of information systems can be seen as a significant system-theoretical attribute. Put simply, this complexity can be attributed to the fact that information systems work on a business, as well as on a technical level. By constructing models, the attempt is made to create manageable artifacts with which the complexity of information systems becomes controllable. The information models created thereby have a tradition of more than thirty years [2; 10; 16]. From today's perspective, these models have established themselves in information systems research as a vital medium for describing operational information systems [18; 20; 23; 28; 34; 40; 41]. The application possibilities inherent in information models range from software

design and the implementation and configuration of standard software to business process reengineering.

Due to the possibility of their reutilization, in many cases the construction of information models is connected to the demand to abstract from enterprise-specific characteristics. Therefore, one differentiates between enterprise-specific information models and reference models. The term “enterprise-specific” characterizes only the individual character of the corresponding information model; there is no restriction to legally independent companies connected with it. For reasons of linguistic clarity it is therefore better to speak of specific information models in order to allow for the fact that the specificity of the models does not result exclusively from the enterprise-context but rather, for example, also from a project-context. To emphasize this context one can also speak, for example, of project-specific models.

In contrast to this, a reference model – in the sense of an initial conceptual approach – is a point of reference for the development of specific models because it represents a category of applications [5, p.90; 35, p.66, pp.69–74; 39, pp.31–38]. Prominent examples of this in the scientific field are the reference model for industrial enterprises (Y-CIM-Model) from SCHEER [32], as well as the SAP R/3-reference model [11] resulting from commercial practice. On the one hand, the possibility of orienting oneself on the technical content of such reference models promises the model-users savings in time and costs, while on the other the quality of the model to be constructed can be increased by the use of a reference model.

Despite these benefits often attributed to reference models in literature, no uniform grasp of the term “reference model” exists. In research and practice different types of models are referred to as reference models. For HARS for example, the term reference model “belongs to a class of terms used often but rarely defined clearly” [17, p.12]. Even a decade after this assertion the situation has barely changed. Although the term reference model was defined more precisely at the end of the 1990ies during the conference *Reference Modeling* – a summary of the conference series is available under the URL <http://www.wi.uni-muenster.de/is/Tagung/> – and the dissertation from SCHÜTTE [35] – that is at least in German-speaking regions – the tendency in literature towards generally declaring information models as reference models still exists. In this respect, the assertion from LEHNER that “in a sense every model can be understood as a reference model” [22, p.126] is not surprising. The question as to why recommended models “warrant” the attribute “reference” in literature often goes unanswered. Here, we have singled out one of the many current unfounded reference model declarations from the German-language information systems community: Using a reference model AHLEMANN, HAAS, HOPPE [3] show how e-learning can be systematically integrated in the further education of companies by establishing it in the operational planning system. Although they explain their grasp of the term reference model according to SCHÜTTE [35, p.69], why they refer to their model as a reference model and not an information model is not explained.

The following analysis on the way the term reference model is understood in information systems research takes this situation into account. It is structured in the following manner: Section 2 first lists “early” considerations to the term reference model from a historic perspective, as well as giving an etymological analysis of the term.

Following this, in Section 3, the current attribute-based characterizations of the term reference model in the literature of today will be discussed critically. The insights resulting from this will flow into a set-theoretic illustration, as well as an explanation of the way the term is understood in Section 4. A critical discussion of the findings in Section 5 shows the consequences resulting from the definition presented here for the use of reference models. The article ends with a conclusion in Section 6.

2 Etymology and History of the Term Reference Model

From an etymological view, the term “reference” has a double significance. In addition to its meaning as a recommendation, the word “reference” is also used in the sense of bearing a relation to something, quoting something or alluding to something. The term “reference” was initially used in the business language of the 19th century to denote a person or company able to give information concerning the trustworthiness of a business partner. The definition of a person or place to whom or where one could appeal for his or her (social) recommendation came later [1, p.464].

In linguistics “reference” also refers to the relationship between linguistic symbols and their contributor in the extra-linguistic reality. In economics, “reference” is used to describe a state which can not be achieved in reality or a state of affairs of exemplary nature. Thus for example, the model of perfect competition, discarded to a large extent due to its restrictive assumptions, is accepted as a reference. In information modeling, one also speaks of a model being consulted as an ideal type of reference object or as a recommendation for the development of other models.

The historic roots of the term reference model in information systems research can only be traced with difficulty. Nevertheless, early clues to the basic idea of reference modeling can be found in the literature which today essentially consists in the systematic structuring and reutilizing of operational tasks for their data processing support.

The significance of graphic models valid for a class of applications was discussed early on in business administration literature. Already 1931 NORDSIECK characterized in *Grundprobleme und Grundprinzipien der Organisation des Betriebsaufbaus* so-called *Aufgabengliederungspläne*¹ as follows: “Usually, a task structuring plan already has a relatively universal character because it is created according to logical principles, i.e. it is not only valid for the company being studied but rather – with a few changes – for companies with similar aims and the same branch of trade” [24, p.160].

Also, the *ideal models* described by KOSIOL in an analysis of the relationships between business administration and operations research come close to today’s term reference model. He explains: “So-called real models which try to represent objects

¹ Today the term “Aufgabengliederungspläne” which may be translated as “task structuring plan” has gone out of use. The technical terms “function hierarchy diagram” resp. “function tree” have won recognition as terms in the meaning of the corresponding modeling language.

of empirical reality are opposed to ideal models which exhibit no reference to reality or leave this open" [21, p.755]. He adds, that "ideal models are the constructs of operations research which represent a larger area of possible real-life situations and serve as prefabricated solutions or standard recipes for certain categories of decision problems in coping with practical problems" [21, p.758].

Another early paraphrase for the fundamental idea of reference modeling can be found in the environment of the *System Dynamics* approach going back to FORRESTER. This is a concept founded on the systems theory for the model-based description and simulation of dynamic systems. In 1968 FORRESTER wrote retrospectively: "A person applying the industrial dynamics approach to actual corporate problems seems to do so by drawing heavily on his mental library of the systems which he has previously studied. If others are to be able to do the same, such libraries of examples must be put in orderly written form. Such a series of structures would identify those relationships which are found repeatedly in industry. [...] Such a treatment of systems should concentrate on the minimum structure necessary to create a particular mode of behavior." [13] FORRESTER thus characterizes an attribute of reference models which attempt to abstract from individual characteristics in order to make themselves reusable.

The question however still exists, as to which origins the term reference model can be traced back to. There is a consensus in literature on the fact that the terminological foundation for "reference model" – in terms of a reference information model – was laid with the *Köln Integration Model* (KIM) [15; 16]. However, neither of these publications speaks of a "reference model". Instead they speak of the development of a "universal model for an integrated data processing system" [16, p.VII], a "basic model" [16, p.X] or a "model template" [15, p.44]. These terms characterize models "that are generalized in a way, that they are not specific to an individual company, but rather characteristic for all resp. the lion's share of companies from a certain group or branch of trade" [15, p.43]. These models should serve in helping companies to create their own individual information system [16, p.X].

Despite these early references to the significance of universal models and their usefulness as templates for the derivation of enterprise-specific models, the technical term "reference model" first established itself in literature towards the end of the 1980ies [14; 26; 29; 43]. This chronological correlation can be supported by looking at different editions of the book *Business Process Engineering* from SCHEER [32]. In the first edition, the data model developed therein is referred to as an integrated database schema resp. an enterprise-wide data model [29]. Then, in the preface of the second edition, SCHEER states that the consideration of the company data model was complemented by practical experience gained in the between-time using the model as a basis for enterprise-specific data models [32, p.VIII]. At another point in the same edition he makes this statement more precise by remarking, that the model had already been used several times as a reference model in setting up enterprise-wide data

models [32, pp. 542ff.].² The acceptance of the model as a reference model in practice even prompted SCHEER to give the book a different subtitle in the second edition *Reference Models for Industrial Enterprises* [32]. This publication was material to the coinage of the term “reference model” – that is, in the realm of German-speaking information systems research.

3 Characterization of the Term Reference Model Based on Attributes

The proposed reference model terms in information systems literature are generally based upon attributes which characterize these reference models, in particular, the attributes “universality” and “recommendation character” [39, pp. 31 ff.].

3.1 The Attribute Universality

The demand for universality as a constituent attribute of the term reference model can be found in many works [5, p. 90; 17, p. 15; 19, p. 12; 35, p. 69; 42, p. 127]. For purposes of simplification these publications also talk of the universality of reference information models. HARS for example, sees the universality of a reference model as a prerequisite for it serving as a source for the creation of a specific model [17, p. 15]. JOST explicitly emphasizes that the character of universality is the most significant attribute of a reference model [19, p. 12]. The fact however, that the universality of a reference model can not be understood in the sense of the model’s claim to absoluteness, i.e. a claim to universal validity, often goes unrecognized. A reference model can only be (universally) valid with regard to a certain category of applications, for example a category of enterprises or a category of projects. Already in 1980 BRETZKE differentiated in his analysis *Der Problembezug von Entscheidungsmodellen* between two types of models, concrete and common decision models [8, pp. 10ff.]. If one transfers his remarks to enterprise-specific models and reference models, then a reference model “is characterized by the fact that it applies to a certain category of situations. It is not universal because it is always valid, but rather because it is always valid under certain circumstances (contained within itself)” [8, p. 11]. To speak of the universality of a reference model is therefore seen as being inexpedient in this article. Thus, the allowance for a corresponding constituent attribute for the term reference model was not considered here.

² Compare this assessment to the statement from ÖSTERLE, BRENNER, HILBERS, who believe that the data models from [29] are primarily to be consulted as reference models for the development of enterprise-specific models [25, p. 71].

3.2 The Attribute Recommendation Character

In addition to universality, it is possible in other works to find the demand for a recommendation character as a constituent attribute for the term reference model [4, pp. 25f.; 5, pp. 86, 90; 7, p. 428; 27, pp. 16f.; 35, p. 69]. Authors connect such a recommendation with the fact that reference models have a standard character for a certain class of applications. They serve as a default solution, from which enterprise-specific concretizations can be derived (economically). Similar to the argumentation in the previous section, the demand for a recommendation character for reference models also proves to be critical. For example, it is unclear how the quality of a recommendation for a reference model can be verified – in this regard VOM BROCKE [39, p. 32] also speaks of the lack of assessability for the content of a recommendation: Which model can be granted or even denied recommendation character subject to which attributes? Which demands can be made on the recommendation or those making the recommendation? These questions make it obvious that this is a question of a non-operational aspect. The user cannot decide upon the recommendation character of a model objectively, but rather only subjectively within the scope of its application. Therefore, this attribute must also be seen as non-constituent for the term reference model in this article.

4 Implications for the Term Reference Model

4.1 Set-Theoretic Illustration of the Way the Term Reference Model is Understood

Since both attributes “universality” and “recommendation character” have been excluded as constituent attributes for the term reference model, the question remains as to how a model becomes a “reference”. To answer this question we must first look at model-theoretic principles [36] in which a developer and a user perspective on models are taken into consideration. Using these perspectives one can discern whether a model is *declared to be* a reference model (developer’s perspective) or whether it is *accepted* as a reference model (user perspective). “Or” is not used here in its colloquial sense, but rather should be understood in a Boolean sense as an adjunction (non-excluding “or”), so that the case of the developer-sided declaration *and* the user-sided acceptance is also taken into consideration. Elementary set-theoretic considerations were consulted in order to illustrate possible situations. These are illustrated in Fig. 1 and will be explained in the following.

The basic set seen in Fig. 1 is the set of all information models IM . As subsets of this set, the set of the information models declared to be reference models by the developers of the models $RM_{\text{Declaration}}$, as well as the set of the information models used by model-users for the construction of specific models $RM_{\text{Acceptance}}$ are plotted. For the characterization of reference models three situations are conceivable:

1. $RM_{\text{Declaration}} \cap \overline{RM_{\text{Acceptance}}}$: The elements of this set are declared as reference models without being accepted by a user. In this case, the property of being a reference model is based upon the assertion of the developers.
2. $\overline{RM_{\text{Declaration}}} \cap RM_{\text{Acceptance}}$: It is conceivable that users consult a model for the construction of specific models, although the model's developers did not initially intend this. Corresponding information models are characterized by this set.
3. $RM_{\text{Declaration}} \cap RM_{\text{Acceptance}}$: The intersection of both sets takes the information models declared to be reference models by the developers, as well as those accepted by the users as such into account. A consensus between developer and user exists in regard to the characterization of the elements of this set as reference models.³

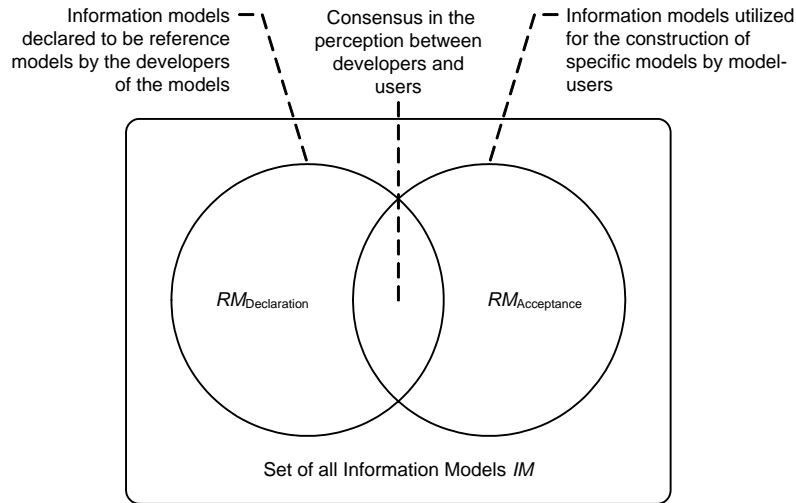


Fig. 1. Set-theoretic illustration of the term reference model

In this article, the developer-sided declaration as reference model (cp. set $RM_{\text{Declaration}}$ in Fig. 1) is seen neither as a necessary, nor as a sufficient criterion for the characterization of a reference model. A developer's assertion that he has constructed a universally valid and recommendable model remains meaningless for the time being. In this context, VOM BROCKE also speaks of "reference character at plan level" [39, p.33, fn. 140]. This attribute can ultimately be proved only by way of the model being applied at least once. SCHEER argues similarly. He concretizes the demands on a reference model from a user's point of view to the effect that at least one application must be conceivable for the use of the model, unchanged, as a specific model [33, p. 4]. In an economic sense, a reference model that goes unused undoubt-

³ This point of view precludes the case that a model seen neither from the developer's side nor from the user-side as a reference can be declared a reference model [39, p.32, fn. 139]. Moreover, it remains unclear in this case whose task it is to make the declaration resp. the model as a reference model.

edly falls short of its basic intention. The use of an information model by a model-user for the derivation of specific models, i.e. its acceptance as a reference (cp. set $RM_{Acceptance}$ in Fig. 1) can thus be seen as a necessary criterion for the characterization of a model as a reference.

To clarify whether it can also be acknowledged as a sufficient criterion, two cases can be distinguished from a set-theoretic point of view. Either the model was also recommended by the developer as a reference, i.e. it is contained in the set $RM_{Declaration} \cap RM_{Acceptance}$ or the developer did not intend this, i.e. it is contained in the set $\overline{RM_{Declaration}} \cap RM_{Acceptance}$ (cp. Fig. 1). The first case can be seen as ideal and thus as uncritical due to the consensus between the developer and user. This paper however, also recognizes the second case as constituent for the term reference model. As a result, only the user can make the decision as to whether a model can be recognized as a reference. User-sided acceptance can be seen as a sufficient criterion. Consequently, it is possible for a model to become a reference model at its initial application; if need be without the knowledge of its developer [39, p.34].

It would be ideal for the constructor of a model to declare his model a reference model only when its application is known to him in at least one case. This grasp of the term is justified by the example from literature already discussed in Section 2. SCHEER also initially developed a data model which he then recommended for the derivation of enterprise-specific models [29]. It has however, turned out that in practice the model's recommendation character was accepted. Thus for example, a field report from BÜRLI et al. [9] was published on the derivation of a specific model based upon the information model from SCHEER for the field of production planning and control. This then prompted SCHEER to declare the model to be a reference model [32].

4.2 Explanation of the Term Reference Model

The term reference model can be explained as a concretion of the term “information model” on the basis of the constituent attribute of user-sided acceptance: A *Reference model* – specifically: reference information model – is an information model used for supporting the construction of other models.

This definition stands in the tradition of early definitions and emphasizes the benefits of reference models “as a fundamental starting point for the development of new information systems” [30, p.94]. HARS also emphasizes the user-sided acceptance by stating that “every reference model is a model which can be consulted for the development of other models” [17, p.15]. SCHEER later abstracts from information models and sees a reference model “as a model which can serve as the starting point for the development of solutions based on concrete problems” [33, p.3]. A corresponding tendency in emphasizing the use of reference models can also be observed in the more recent literature of reference modeling. Thus BECKER, KNACKSTEDT refer to information models used as initial solutions for the development of project-specific models [6, p.415], as reference models.

Consequently, the author pleads for a use-oriented reference model term. Every model resp. partial model which can be used in supporting the construction of another model can be seen in this sense as a reference model. The reutilization of reference models connected with this can be seen as a fundamental idea resulting from the paperless, tool-supported data-processing consulting at the beginning of the 1990ies [31] and must be emphasized as a fundamental characteristic of reference models.

5 Discussion on the Term Reference Model as Defined Here

5.1 Consequences for Reference Modeling Research

Studies in the field of reference modeling must often deal with the fundamental problem of finding and locating reference models. Because reference models are understood as special information models, the search can initially be limited to information models. When an information model is found one must then decide whether it is a case of a reference model or not. In making this decision, the person searching for the model is confronted with a problem in two respects. Firstly, one can only subjectively decide whether a model is a reference model. However, even if one person accepts a reference model as such, this does not mean that the next person will also do so. And secondly, identifying criteria such as universality or recommendation character must be dismissed as constituent characteristics of a reference model. This examination follows the use-oriented reference model term from Section 4, which is directed at the model's use. The models declared exclusively as being reference models are not accepted as such.

Were one to use this "restrictive" understanding of the term in reference modeling research the number of actual reference models would be small, because by close interpretation the existence of at least one application – moreover: its documentation – would be essential. The focus of contextual studies in reference modeling should therefore be extended to the models only declared as being reference models. This corresponds in two ways with the pragmatic orientation expounded upon at the beginning of this article. On the one hand, the topic of this article is not to judge whether models declared to be reference models in literature should actually be accepted as such. And on the other, the deduction of future research guidelines can only be possible by way of analyzing prevailing perceptions.

5.2 Consequences for the Management of Reference Models

It is irrelevant for a model's user – in the sense of the term reference model used in this paper – whether a model, whose content he wishes to reuse, has been recommended for use by the model's developer i.e. was declared to be a reference model or not. He orients his decision on the use of a reference model only on whether he can recognize a potential benefit from the model. In order to make this decision the reference model must be made available to the user. An important prerequisite for the

structuring of this availability is the systematic management of the current stock of reference models [37; 38].

Despite the variety of existing reference models there are very few studies in literature with the verification and documentation of actual reference models as their subject. Based on this, there is also a lack of studies regarding the question of which reference models should be used in which situations. A very small number of approaches deal with the systematization of reference models, whereby it is in fact the tabulation of reference models that is meant here and not so much the survey-like textual description of the actual stock of reference models found in literature. The most comprehensive results were delivered by the analyses from FETTKE and LOOS [12] on the catalog-based reutilization of reference models in which the authors transfer the concept of a construction catalog used in engineering to reference modeling. These so-called *reference model catalogs* represent without a doubt, a meaningful tool for the systematic management of reference models.

However, it must be pointed out with respect to the cataloging only of reference information models, that those involved in the development and administration of a reference model catalog also have the problem mentioned in Section 5.1: They must decide which information model can be accepted as a reference model and thus be cataloged. Taking into consideration the exact interpretation of the term reference model which is the basis of this article, only models for which at least one application exists could then be cataloged. This implicates that the users of such a catalog, limited only to reference models, would be principally refused access to enterprise-specific models. This circumstance contradicts the pragmatic focus of the use-oriented reference model term in this study, because it remains unconsidered that information models generally – even when the developer has declared it a reference model or it has already been used – are used to support the construction of other models. This results, for the design of a reference model catalog, in the need for an expansion in the direction of a systematic organization of information models, independent of their contextual individuality.

5.3 Consequences for the Creation of Reference Modeling Languages

The use-oriented reference model term underlined in this publication emphasizes the use of a reference model for the construction of enterprise-specific models. The user's task during construction, which can be supported by IT tools, consists in the adaptation of the reference model. In a figurative sense, the derivation of specific models from a reference model characterized by this term is equivalent to the creation of different variants of the reference model [35, pp.207–209]. Thus, for example, the enterprise-specific models *information model product-oriented Manufacturing Enterprise* E_1 or *information model process-oriented Manufacturing Enterprise* E_2 could be derived as variants of the reference model *Manufacturing*.

The management of variants derived from reference models is especially interesting in two respects. Firstly, the storage of the variants in connection with the adaptation-premises also administrated can speed up the future development of enterprise-

specific models for comparable applications. Secondly, this also allows for a similarity analysis of the variants whose results can then be used for the development of new reference models.

Reference modeling languages must therefore be created so that they support model-variant management. However, contradictory opinions exist in literature as to which construction technique should be used for reference model-variant management. While for example, SCHÜTTE ties variant management ex ante to the construction technique of the configuration [35, pp.207ff.] and also refers to a variant as a configured output in his terminology [35, p.207, fn.91], VOM BROCKE [39, p.101] argues against the coupling of variant management with individual construction techniques and proposes further construction techniques with aggregation, specialization, instantiation, and analogy construction [39, pp.235ff.]. These construction techniques for the adaptation of models must be embedded in modeling languages. The effort needed for the expansion of these languages is however, so high that it can, by all means, more than make up for the benefits which can be achieved by adapting reference models within the framework of modeling projects. Reference modeling research must therefore dedicate itself more heavily to the question of profitability in the application of reference models in the future.

6 Final Conclusion

The topic of this article was the detailed analysis of the understanding of reference models in the information systems discipline. The author did not intend to create a comprehensive and universally valid definition of the term. In fact, the author's aim was to examine the term "reference model" from different perspectives and on the basis of this, create an understanding which he hopes will prove to be useful in the context of information systems research. The author hopes to have contributed a valuable share in answering the question "What is a reference model?" put in the call of the *Workshop on Business Process Reference Models* (BPRM 2005).

The need clearly remains for more fundamental research in order to understand the effects connected to the creation and use of reference models in research and practice. However reference modeling research approaches this topic in the future, the compilation of improved knowledge on the application systems and organizations remains a central topic in this field of research. A terminological foundation for the management of this knowledge represented by reference models has been made available by the insights gained within the scope of this article.

Acknowledgement. This paper presents results from the research project "Reference Model-Based Customizing with Vague Data", abbreviated "Fuzzy-Customizing", funded by the Deutsche Forschungsgemeinschaft (German Research Foundation) as part of the initiative BRID².

References

1. Drosdowski, G. (ed.): *Duden Etymologie : Herkunftswörterbuch der deutschen Sprache*. 2nd ed. Mannheim : Dudenverl., 1989 (in German)
2. Abrial, J.-R.: Data Semantics. In: Klimbie, J. W.; Koffeman, K. L. (eds.): *Data Base Management : Proceeding of the IFIP Working Conference Data Base Management, Cargèse, Corsica, France, 1–5 April, 1974*. Amsterdam : North-Holland Pub. Co, 1974, pp. 1–60
3. Ahlemann, F.; Haas, C.; Hoppe, U.: Organisationale Integration von E-Learning in Unternehmen – ein Referenz-Informationsmodell. In: Uhr, W.; Esswein, W.; Schoop, E. (eds.): *Wirtschaftsinformatik 2003 : Medien – Märkte – Mobilität ; Band 1*. Heidelberg : Physica, 2003, pp. 707–726 (in German)
4. Becker, J.; Delfmann, P.; Knackstedt, R.; Kuropka, D.: Konfigurative Referenzmodellierung. In: Becker, J.; Knackstedt, R. (eds.): *Wissensmanagement mit Referenzmodellen : Konzepte für die Anwendungssystem- und Organisationsgestaltung*. Heidelberg : Physica, 2002, pp. 25–144 (in German)
5. Becker, J.; Holten, R.; Knackstedt, R.; Schütte, R.: Referenz-Informationsmodellierung. In: Bodendorf, F.; Grauer, M. (eds.): *Verbundtagung Wirtschaftsinformatik 2000*. Aachen : Shaker, 2000, pp. 86–109 (in German)
6. Becker, J.; Knackstedt, R.: Konstruktion und Anwendung fachkonzeptioneller Referenzmodelle im Data Warehousing. In: Uhr, W.; Esswein, W.; Schoop, E. (eds.): *Wirtschaftsinformatik 2003 : Medien – Märkte – Mobilität ; Band 2*. Heidelberg : Physica, 2003, pp. 415–434 (in German)
7. Becker, J.; Schütte, R.: Referenz-Informationsmodelle für den Handel: Begriff, Nutzen und Empfehlungen für die Gestaltung und unternehmensspezifische Adaption von Referenzmodellen. In: Krallmann, H. (ed.): *Wirtschaftsinformatik '97 : Internationale Geschäftstätigkeit auf der Basis flexibler Organisationsstrukturen und leistungsfähiger Informationssysteme*. Heidelberg : Physica, 1997, pp. 427–448 (in German)
8. Bretzke, W.-R.: *Der Problembezug von Entscheidungsmodellen*. Tübingen : Mohr, 1980 (in German)
9. Bürli, A.; Jaccottet, B.; Knolmayer, G. F.; Myrach, T.; Küng, P.: Vorgehen beim Aufbau von CIM-Datenmodellen. In: *io Management Zeitschrift* 61 (1992), no. 12, pp. 82–86 (in German)
10. Chen, P. P.-S.: The entity-relationship model – toward a unified view of data. In: *ACM Transactions on Database Systems* 1 (1976), no. 1, pp. 9–36
11. Curran, T. A.; Keller, G.; Ladd, A.: *SAP R/3 business blueprint : Understanding the business process reference model*. Upper Saddle River, NJ : Prentice Hall PTR, 1998
12. Fettke, P.; Loos, P.: Classification of Reference Models – A Methodology and its Application. In: *Information Systems and e-Business Management* 1 (2003), no. 1, pp. 35–53
13. Forrester, J. W.: Industrial Dynamics – After the First Decade. In: *Management Science* 14 (1968), no. 7, pp. 398–415
14. Gersting, J.; Kinsley, K.; McDonald, N.; North, J.; Sastry, M.; Stull, E.: Reference model for DBMS user facility. In: *ACM SIGMOD Record* 17 (1988), no. 2, pp. 23–52
15. Grochla, E.: Das Konzept des Kölner Integrationsmodells. In: Grochla, E. (ed.): *Integrierte Gesamtmodelle der Datenverarbeitung : Entwicklung und Anwendung des Kölner Integrationsmodells (KIM)*. München : Hanser, 1974, pp. 35–46 (in German)
16. Grochla, E.; Garbe, H.; Gillner, R.; Poths, W.: Grundmodell zur Gestaltung eines integrierten Datenverarbeitungssystems : Kölner Integrationsmodell (KIM). In: Grochla, E.; Szyperski, N. (eds.): *Arbeitsberichte des Betriebswirtschaftlichen Instituts für Organisation und Automation (BIFOA) an der Universität zu Köln*, no. 71/6, Köln : WISON Verl., 1971 (in German)

17. Hars, A.: *Referenzdatenmodelle : Grundlagen effizienter Datenmodellierung*. Wiesbaden : Gabler, 1994 (in German)
18. Hay, D. C.: *Requirements analysis : From business views to architecture*. Upper Saddle River, NJ : Prentice Hall PTR, 2003
19. Jost, W.: *EDV-gestützte CIM-Rahmenplanung*. Wiesbaden : Gabler, 1993 (in German)
20. Kilov, H.: *Business models : A guide for business and IT*. Upper Saddle River : Prentice Hall, 2002
21. Kosiol, E.: Betriebswirtschaftslehre und Unternehmensforschung : Eine Untersuchung ihrer Standorte und Beziehungen auf wissenschaftstheoretischer Grundlage. In: *Zeitschrift für Betriebswirtschaft* 34 (1964), no. 12, pp. 743–762 (in German)
22. Lehner, F.: Modelle und Modellierung. In: Lehner, F.; Hildebrand, K.; Maier, R. (eds.): *Wirtschaftsinformatik : Theoretische Grundlagen*. München : Hanser, 1995, pp. 73–164 (in German)
23. Mylopoulos, J.: Information Modeling in the Time of the Revolution. In: *Information Systems* 23 (1998), no. 3/4, pp. 127–155
24. Nordsieck, F.: Grundprobleme und Grundprinzipien der Organisation des Betriebsaufbaus. In: *Die Betriebswirtschaft* 24 (1931), no. 6, pp. 158–162 (in German)
25. Österle, H.; Brenner, W.; Hilbers, K.: *Unternehmensführung und Informationssystem : Der Ansatz des St. Galler Informationssystem-Managements*. 2nd ed. Stuttgart : Teubner, 1992 (in German)
26. Peckham, J.; Maryanski, F.: Semantic data models. In: *ACM Computing Surveys* 20 (1988), no. 3, pp. 153–189
27. Rosemann, M.; Schütte, R.: Grundsätze ordnungsmäßiger Referenzmodellierung. In: Becker, J.; Rosemann, M.; Schütte, R. (eds.): *Entwicklungsstand und Entwicklungsperspektiven der Referenzmodellierung : Proceedings zur Veranstaltung vom 10. März 1997*. Münster : Institut für Wirtschaftsinformatik, Westfälische Wilhelms-Universität, 1997, pp. 16–33 (in German)
28. Rossi, M.; Siau, K.: *Information Modeling in the new Millennium*. Hershey : Idea Group Publishing, 2001
29. Scheer, A.-W.: *Enterprise-wide data modelling : Information systems in industry*. 1st ed. Berlin : Springer, 1989
30. Scheer, A.-W.: Unternehmensdatenmodell. In: *Information Management* 5 (1990), no. 1, pp. 92–94 (in German)
31. Scheer, A.-W.: Papierlose Beratung – Werkzeugunterstützung bei der DV-Beratung. In: Scheer, A.-W. (ed.): *Veröffentlichungen des Instituts für Wirtschaftsinformatik*, no. 81, Saarbrücken : Universität des Saarlandes, 1991 (in German)
32. Scheer, A.-W.: *Business Process Engineering : Reference Models for Industrial Enterprises*. 2nd ed. Berlin : Springer, 1994
33. Scheer, A.-W.: ARIS – House of Business Engineering: Konzept zur Beschreibung und Ausführung von Referenzmodellen. In: Becker, J.; Rosemann, M.; Schütte, R. (eds.): *Entwicklungsstand und Entwicklungsperspektiven der Referenzmodellierung : Proceedings zur Veranstaltung vom 10. März 1997*. Münster : Institut für Wirtschaftsinformatik, Westfälische Wilhelms-Universität, 1997, pp. 3–15 (in German)
34. Scheer, A.-W.: *ARIS – business process modeling*. 2nd ed. Berlin : Springer, 1999
35. Schütte, R.: *Grundsätze ordnungsmäßiger Referenzmodellierung : Konstruktion konfigurations- und anpassungsorientierter Modelle*. Wiesbaden : Gabler, 1998 (in German)
36. Stachowiak, H.: *Allgemeine Modelltheorie*. Wien : Springer, 1973 (in German)
37. Thomas, O.: Reference Model Management. In: *6th International Conference The Modern Information Technology in the Innovation Processes of the Industrial Enterprises MITIP 2004 : September 9–10, 2004 Prague, Czech Republic ; Proceedings*. Prague : Czech Technical University, 2004, pp. 33–36

38. Thomas, O.; Adam, O.; Seel, C.: SAP Business Process Model Management. In: *Business Process Innovation : Proceedings ; SAP Innovation Congress Americas '04, February 28-March 1, 2004*. Orlando : SAP AG, 2004
39. vom Brocke, J.: *Referenzmodellierung : Gestaltung und Verteilung von Konstruktionsprozessen*. Berlin : Logos, 2003 (in German)
40. Wand, Y.; Weber, R.: Research Commentary: Information Systems and Conceptual Modeling – A Research Agenda. In: *Information Systems Research* 13 (2002), no. 4, pp. 363–376
41. Weber, R.: *Ontological foundations of information systems*. Melbourne : Coopers & Lybrand and the Accounting Association of Australia and New Zealand, 1997
42. Wolf, S.: *Wissenschaftstheoretische und fachmethodische Grundlagen der Konstruktion von generischen Referenzmodellen betrieblicher Systeme*. Aachen : Shaker, 2001 (in German)
43. Wollnik, M.: Ein Referenzmodell des Informations-Managements. In: *IM Information Management* 3 (1988), no. 3, pp. 34–43 (in German)

Towards a Reference Model for Work Distribution in Workflow Management Systems

M. Pesic and W.M.P. van der Aalst

Department of Technology Management, Eindhoven University of Technology,
P.O.Box 513, NL-5600 MB, Eindhoven, The Netherlands.
`m.pesic@tm.tue.nl`, `w.m.p.v.d.aalst@tm.tue.nl`

Abstract. Reference models such as the well-known SAP reference models tend to focus on the control-flow perspective. Although the languages typically used to capture reference models (e.g., EPCs) allow for the modeling of the resource or data perspectives, reference models tend to oversimplify these other perspectives. This paper focusses on the *resource perspective* in the context of workflow management systems. The aim is to develop a reference model for work distribution, i.e., how should the system distribute work based on the structure of the organization, capabilities/qualifications of people, and characteristics of the process. This paper reports on our first results based on a detailed analysis of contemporary workflow management systems (Staffware, FileNet, and FLOWer), supported by *Colored Petri Nets* (CPNs) to model work distribution mechanisms and *resource patterns* to identify key functionalities.

Key words: Work distribution, reference models, workflow management, business process management, resource patterns, colored Petri nets.

1 Introduction

Reference models are generic conceptual models that formalize recommended practices for a certain domain. Often labelled with the term “best practice” reference models claim to capture reusable state-of-the-art practices. Reference models typically focus on a specific application domain. For example, The Dutch NVVB (<http://www.nvvb.nl/>) offers a set of reference models for local governments modeled using the Petri-net-based tool Protos [31]. Other reference models are more general, moreover, the term reference model is also used for models describing the structure and functionality of business applications. One could argue that the SAP reference model actually describes the R/3 system rather than “best practices” in some domain. We will interpret reference models in the more system-oriented sense. However, instead of building a system-specific reference model, we would like to generalize over a range of systems, i.e., existing and future workflow management systems.

Workflow management systems are process-aware information systems [1, 12], which are used in companies as a means for the computerized structuring

and driving of complex business processes. Workflow management systems implement business process models and use them for driving the flow of work by allocating the right employees to the right tasks at the right times. The system *manages the work of employees*. It will determine which tasks an employee has to execute and when, which documents will be used, which information will be available during work, etc. Typically, a workflow management system offers several mechanisms to distribute work. Nevertheless, we believe that existing systems are too limited in this respect. The goal of this paper is not to propose advanced work distribution mechanisms. Instead we focus on the analysis of functionality in existing systems. The goal is not to evaluate these systems, but to understand *how* they offer specific functionality. Since work distribution defines the quality of work, it is important to consider research from the field of social sciences, e.g., social-technical design [7, 10, 13, 43]. We believe that only by combining both *technical* and *social* approaches, one can truly grasp certain phenomena. A *deeper understanding* of particular aspects of work distribution is essential for developing a new breed of more user-centric systems.

The work reported in this paper can be seen as an extension of the *workflow patterns initiative* [2] (cf. www.workflowpatterns.com). Within the context of this initiative 43 resource patterns [37, 39] have been defined. Using a patterns approach, work distribution is evaluated from the perspective of the end-user as a dynamic property of workflow management systems. The work reported in this paper adds to a better understanding of these mechanisms by providing explicit process models for these patterns, i.e., the descriptive models are augmented with executable models. Note that most work reported in literature (cf. Section 5) uses static models to describe work distribution. Consider for example the meta modeling approaches presented in [3, 27–29, 36]. These approaches use static models (e.g., UML class diagrams) to discuss work distribution concepts. This paper takes a truly dynamic model – a *Colored Petri Net* model – as a starting point, thus clearly differentiating our contribution from existing work reported in literature.

Colored Petri Nets (CPNs) [18, 23] are a natural extension of the classical Petri net [33]. There are several reasons for selecting CPNs as the language for modeling work distribution in the context of workflow management. First of all, CPNs have formal semantics and allow for different types of analysis, e.g., state-space analysis and invariants [19]. Second, CPNs are executable and allow for rapid prototyping, gaming, and simulation. Third, CPNs are graphical and their notation is similar to existing workflow languages. Finally, the CPN language is supported by CPN Tools¹ – a graphical environment to model, enact and analyze CPNs.

In this paper, we provide a basic CPN model that can be seen as the “greatest common denominator” of existing workflow management systems. The model will incorporate concepts of task, case, user, work item, role and group. This model should be seen as a *starting point* towards a more *comprehensive reference model for work distribution*. The basic CPN model is extended and specialized

¹ CPN Tools can be downloaded from wiki.daimi.au.dk/cpntools/.

for three specific systems: Staffware [42], FileNet [15], and FLOWer [30]. The latter three models are used to investigate differences and similarities as aid in a deeper understanding of work distribution mechanisms. In addition, advanced resource patterns that are not supported by these three systems are modeled by extending the basic CPN model.

The remainder of this paper is organized as follows. Section 2 discusses the various types of reference models and how our work can be positioned in a wider range of reference models. Section 3 presents the basic CPN model which should be considered as the “greatest common denominator” of existing workflow management systems. Section 4 extends this model in two directions: (1) Section 4.1 discusses the model in the context of three different systems (i.e., Staffware, FileNet, and FLOWer), and (2) Section 4.2 reflects on the basic model from the perspective of the so-called “resource patterns”. An overview of related work is given in Section 5. Section 6 concludes the paper.

2 Reference Models

As indicated in the introduction, we can distinguish at least two types of reference models: (1) “best practice” reference models that aim at capturing domain-specific practices, and (2) “system oriented” reference models that aim at capturing the structure and functionality of a software system [9]. Although the focus of this paper is on the latter class of reference models, we first discuss characteristics of reference models in a broader context.

The main objective of reference models is to streamline the design of particular models by providing a generic solution [35]. The application of reference models is motivated by the “Design by Reuse” paradigm. Reference models accelerate the modeling process by providing a repository of potentially relevant models. These models are ideally “plug and play” but often require some customization/configuration [6]. Reference models can be differentiated along the following main criteria [35]: scope of the model (e.g., functional areas covered), granularity of the model (e.g., number of levels of decomposition detail), views (e.g., process, data, objects, organization) that are depicted in the model, degree of integration between the views, purposes supported, user groups addressed, internal or external (commercial) use, availability of the model (e.g., paper, tool-based, Web-based), availability of further textual explanation of the model, explicit inclusion of alternative business scenarios, existence of guidelines on how to use these models, and availability of relevant quantitative benchmarking data. A further and more comprehensive differentiation based upon the domain that underlies the reference model can be found in [5, 9, 21, 34]. In this paper, we look at a reference model focusing on the resource perspective (i.e., the scope is work distribution and the view is the interaction between the process and the organization) at a finer level of granularity.

One of the most comprehensive models is the SAP reference model [9, 21]. Its data model includes more than 4000 entity types and the reference process models cover more than 1000 business processes and inter-organizational busi-

ness scenarios [35]. Most of the other dominant ERP vendors have similar or alternative approaches towards reference models. Foundational conceptual work for the SAP reference model had been conducted in the years 1990-1992 [20]. The outcome of this project was the process modeling language Event-driven Process Chains (EPCs) [20, 22], which has been used for the design of the reference process models in SAP. EPCs also became the core modeling language in the Architecture of Integrated Information Systems (ARIS) [40, 41]. It is now one of the most popular reference modeling languages and has also been used for the design of many SAP-independent reference models (e.g., the ARIS-based reference model for Siebel CRM or industry models for banking, retail, insurance, telecommunication, etc.).

Reference models such as the SAP reference model provide for modelling various perspectives (e.g., the process perspective, the data perspective, etc.). However, existing languages for representing reference models (e.g., EPCs and UML activity diagrams) tend to oversimplify the resource/organizational perspective. When it comes to work distribution there are subtle but very important differences between mechanisms. A badly chosen work distribution mechanism may be very disruptive, and have dramatic effects on the performance of a business process. In the remainder, we will investigate the possibility of a comprehensive CPN-based reference model to overcome these problems.

3 Towards a Reference Model for Work Distribution

Different workflow management systems tend to use not only different work distribution concepts, but also completely different terminologies. This makes it difficult to compare these systems. Therefore, we will not start by developing CPN models for different systems and see how these can be unified, but, instead, start with modeling the “greatest common denominator” of existing systems. This model can assist in comparing systems and unifying concepts and terminology. We will use the term *Basic Model* to refer to this “greatest common denominator” and represent it in terms of a CPN model.

In the introduction we already motivated the use of CPNs as a modeling language [18, 23]. A CPN consists of *places* and *transitions* connected by *arcs*. The network structure is static but places can hold *tokens* thus representing the state of the model. The number of tokens per place can vary over time. Moreover, unlike the classical Petri net, tokens can have both a value and a timestamp. The timestamps indicate the availability of tokens and can be used to model delays, processing times, timeouts, etc. The value of a token indicates the properties of the object represented by this token. Places (represented by ovals) are typed, i.e., the tokens in a place have values of a particular type (or color in CPN jargon). These types are a subset of the data types in Standard ML such as the primitive types integer and string and compositional types such as tuple, list and record. Each place can hold tokens with values of a certain type. Transitions (represented by rectangles) may consume and produce tokens. Since tokens have values, *arc inscriptions* are needed to specify the input-output relations.

Besides the extension with token colors and timestamps, CPN models allow for hierarchy. Complex models may be decomposed into subpages, also referred to as subprocesses or modules, to obtain a layered hierarchical description. A more detailed discussion of the CPN concepts is beyond the scope of this paper. In the remainder, we assume that the reader is familiar with the CPN language and refer to [18, 23] for more details.

The Basic Model represents a workflow management system that enables the following concepts: The business *process* is defined as a set of *tasks*. Before the process can be executed, it has to be instantiated. One (executable) instance of a process is referred to as a *case*. Each case traverses the process. If a task is enabled for a specific case, a *work item*, i.e., a concrete piece of work, is created. There is a set of *users* that can execute work items. The users are embedded in the organizational structure on the basis of their *roles*, and the *groups* they belong to. A group is an organizational unit (e.g., ‘sales’, ‘purchasing’, ‘production’, etc.), while a role represents a capability of the user (e.g., ‘manager’, ‘software developer’, ‘accountant’, etc.). These concepts are mapped onto CPN types as shown in Table 1. As indicated, CPN uses Standard ML types (e.g., *string* and *int*) and type constructors such as *product* to create pairs and other complex constructs (e.g., $(1, "taskA")$ represents a value of type *WI*).

During the distribution, work items change state, which determines the next actions the users and the distribution mechanism can perform. The Basic Model uses a simple model of the life cycle of work items as shown in Figure 1. After the *new* work item has arrived, it is assumed that it is also *enabled* in order to be taken into distribution (i.e., state *initiated*). The Basic Model assumes that a work item becomes enabled at the moment of creation (arrival). Next, the work item is *offered* to the user(s). Once a user *selects* the work item, it is *assigned* to him/her, and he/she can *start executing* it. After the execution, the work item is considered *completed*, and the user can continue working on the next work item. Note that this description covers only the general, rather simplified, behavior of workflow management systems (e.g., errors and aborts are not considered).

Before starting the model, it is necessary to provide the description of a concrete situation that is to be executed. This is done by defining the value of input elements as shown in Table 2.

```

color Task = string;
color Case = int;
color User = string;
color WI = product Case * Task;
color Role = string;
color Group = string;

```

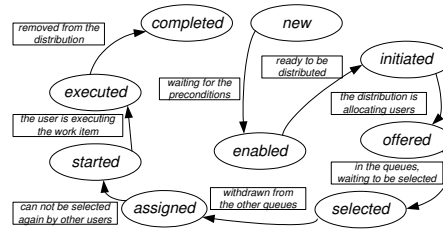


Table 1. Basic Workflow Concepts **Fig. 1.** Basic Model - Life Cycle of a Work Item

Table 2. Input For The Basic Model

| name | color | description |
|----------------|---|---|
| new work items | color WI = product Case * Task; | work items that have arrived and are ready to be distributed to users; |
| system users | color Users = list User; | a set of available users; |
| task maps | color TMap = product Task * Role * Group; | decision about which work items can be executed by which users is made based on the authorizations given in the process definition, for every task; |
| user maps | color UMap = product User * Roles * Groups; | the organizational structure is used to map users to the authorization of tasks; |

As a model of an abstract workflow management system, the Basic Model is made on the basis of predefined assumptions: (1) we abstract from the process perspective (i.e., splits, joins, creation of work items), (2) we only consider the “normal” behavior (i.e., work items are completed successfully; errors and aborts are not included), and (3) we abstract from the user interface.

The model structure is organized into two sub-systems as shown in Figure 2. The CPN language allows for the decomposition of complex nets into subpages. These subpages are also referred to as sub-systems, sub-processes or modules. Using such modules we obtain a layered hierarchical description.

The two modules shown in Figure 2 communicate by exchanging messages via six places. The messages contain information about a user and a work item. Each of the six message places is of the type *color UWI = product User * WI*, i.e., each token represents a “user work item” – a combination of a work item and a user (cf. Table 3).

Work Distribution. Figure 3(a) shows the Work Distribution module. This module manages the distribution of work items. It allocates users to which the work items should be offered, based on authorization (*TMap*) and organization (*UMap*) data. It should also manage the process of work execution, and make sure that work items are executed correctly. The variables used in this module are shown in Table 4.

Table 3. Messages Between Modules (All of type *color UWI = product User * WI*)

| Place | Message |
|------------------------|--|
| <i>to be offered</i> | The work item is offered to the user. |
| <i>withdrawn offer</i> | Withdraw the offered work item from the user. |
| <i>selected</i> | The user requests to select the work item. |
| <i>approved</i> | Allow the user to select the work item. |
| <i>rejected</i> | Do not allow the user to select the work item. |
| <i>completed</i> | The user has completed executing the work item |

The allocation function *offer* contains allocation rules of the specific distribution mechanism. Work items that are offered to users are stored in the place *offered work items*. After receiving a request from the user to select the work item, the decision is made whether to allow the user to select the item (and thus to execute it), or to reject this request. This decision is based on the assumption that at one moment, only one user can work on the work item. If the work item has already been selected (i.e., it is not in the place *offered work items*), then the request is rejected. Otherwise, the approval is sent to the user and the work item is moved to the place *assigned work items*, and, therefore, it cannot be selected again.

Work Lists. Figure 3(b) shows the Work Lists module. This module receives messages from the Work Distribution module about which work items are to be offered to which users. The Work Lists module further manages events associated with the activities of users. It is decomposed into three units, which correspond to three basic actions users can make: *log on and off* (cf. Figure 3(c)) in the system, *select work* (cf. Figure 3(d)), *start work* (cf. Figure 3(e)), and *stop work* (cf. Figure 3(f)). Once the work item has been *offered* to users, they can *select* it. When a user selects the work item, the *request* is sent to the Work Distribution module. If the request is *rejected*, the action is *aborted*. If the Work Distribution Module *approves* the request, the user can start working on the work item. Once the user has started working, the work item is considered to be in *progress*. Next, the user can stop working, and the work item is *completed*. In order to perform any of these actions, it is necessary that the user is *logged on* in the system.

4 Evaluation of the Basic Model

The Basic Model presented in the previous section is used as a kind of reference for different extensions and specializations of work distribution. We have extended and specialized the Basic Model for three concrete systems (Staffware, FileNet and FLOWer) [32]. In this section, we evaluate the basic model by discussing differences between and commonalities among Staffware, FileNet, FLOWer and the Basic Model. Moreover, in Section 4.2, we discuss the relation between the resource patterns reported in [37, 39] and the Basic Model.

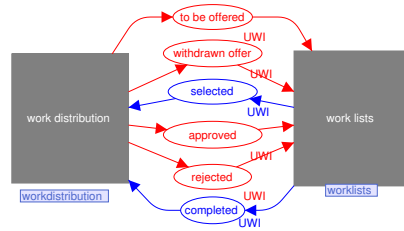


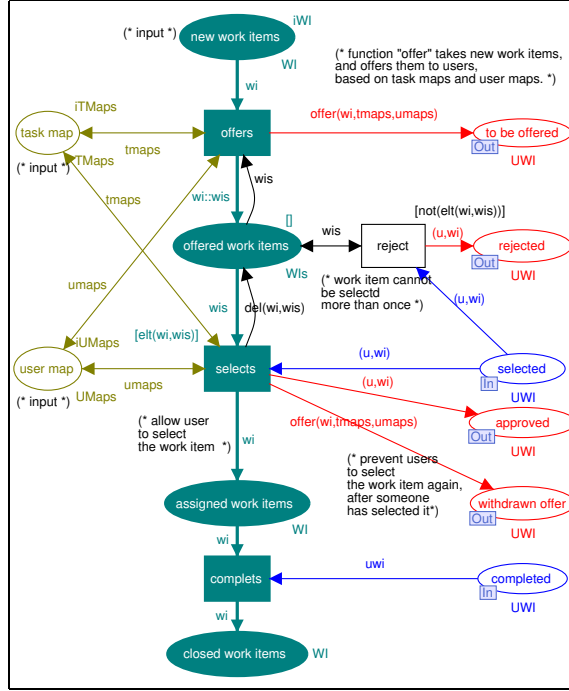
Fig. 2. Basic Model - Work Distribution

Table 4. Basic Model - Variables in Work Distribution Module

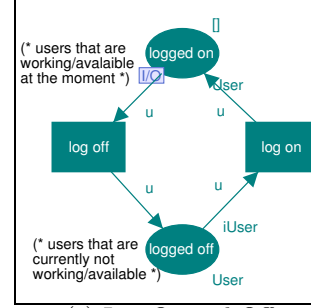
```

var tmaps: TMaps;
var umaps: UMaps;
var wi: WI;
var wis: WIs; ( color WIs = list WI; )
var uwi: UWI;

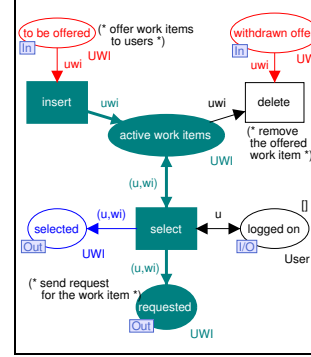
```



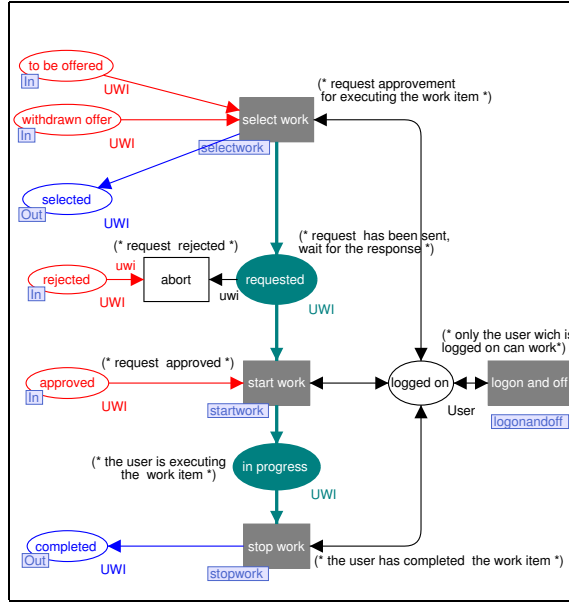
(a) Work Distribution



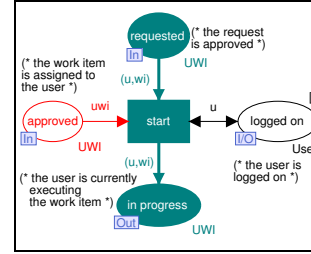
(c) Log On and Off



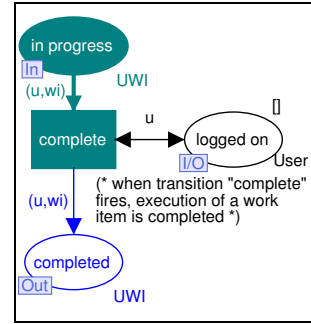
(d) Select Work



(b) Work Lists



(e) Start Work



(f) Stop Work

Fig. 3. Basic Model

4.1 Three Workflow Management Systems: Staffware, FileNet and FLOWer

Staffware and FileNet are two “traditional” workflow management systems. FLOWer can be characterized as a case handling system [4] allowing for more flexibility. We have developed three dedicated CPN models for these three systems. We are not able to show these models here and need to refer to a technical report of this [32]. However, we are able to report on our experiences.

To model the functionality of Staffware, the concept of *work queues* is introduced in the CPN model. In Staffware there are personal queues and group queues. If the same work item is offered to *multiple* work queues, it is executed multiple times. Staffware also allows for *allocation at run-time* based on the attributes of a case. Moreover, Staffware also allows for *forward* and *suspend*, i.e., a user can put a work item on hold (suspend) or forward it to the another user.

FileNet allows for two ways of grouping people in organizational entities: *work queues* and *workflow groups*. Similar to Staffware, FileNet allows for personal queues and group queues. Workflow groups offer a completely different functionality: multiple people can work on the same work item and the group structure may change at run-time. Similar to Staffware, FileNet allows for *forward* and *suspend*.

FLOWer is quite different from Staffware and FileNet because it is data-driven and allows for all kinds of case-handling functionality [4]. This implies several extensions of the Basic Model. FLOWer allows users to *skip* or *redo* activities in addition to simply executing them. Unlike most systems, FLOWer *separates authorization* (“can do”) from *distribution* (“should do”).

Since we cannot show the full CPN models, we limit ourselves to some general conclusions. First of all, the Basic Model is a good basis for building system-specific models. The extensions are typically straightforward and consistent with the core structure of the Basic Model. Second, systems have surprisingly strange limitations, e.g., Staffware supports the role concept, but roles need to be associated to a single user (i.e., no two users can have the same role). Third, systems offer very different functionalities when it comes to work distribution. Finally, in each of the systems the basic concepts are presented and named differently, although a close observation often shows that these system-specific concepts are actually identical.

4.2 Resource Patterns

Instead of extending the Basic Model for more systems, we also looked at a more systematic way of work distribution. As indicated, similar concepts are often named and presented differently. Therefore, it is interesting to define these concepts in a system-independent manner. Therefore, we have used 43 documented *resource patterns* [37, 39]. These patterns are grouped into a number of categories: *creation patterns*, *push patterns*, *pull patterns*, *detour patterns*, *auto-start patterns*, *visibility patterns*, and *multiple resource patterns*. Each of these patterns can be modeled in terms of a CPN model.

Table 5. Support for Resource Patterns in 3 Workflow Systems and Basic Model

(+ = direct support, - = no direct support, +/- = partial support, o = out-of-scope)

| Nr | Pattern | SW | FN | FW | BM |
|----|--|-----|-----|-----|-----|
| 1 | Direct Allocation | + | + | + | +/- |
| 2 | Role-based Allocation | + | +/- | + | + |
| 3 | Deferred Allocation | + | + | - | - |
| 4 | Authorization | - | - | + | - |
| 5 | Separation of Duties | - | - | + | - |
| 6 | Case Handling | - | - | + | - |
| 7 | Retain Familiar | - | - | + | - |
| 8 | Capability-based Allocation | - | - | + | - |
| 9 | History-based Allocation | - | - | - | - |
| 10 | Organizational Allocation | +/- | +/- | +/- | +/- |
| 11 | Automatic Execution | + | + | + | o |
| 12 | Distribution by Offer – Single Resource | - | - | - | - |
| 13 | Distribution by Offer – Multiple Resources | + | + | + | + |
| 14 | Distribution by Allocation – Single Resource | + | + | + | - |
| 15 | Random Allocation | - | - | - | + |
| 16 | Round Robin Allocation | - | - | - | - |
| 17 | Shortest Queue | - | - | - | - |
| 18 | Early Distribution | - | - | + | - |
| 19 | Distribution on Enablement | + | + | + | + |
| 20 | Late Distribution | - | - | - | - |
| 21 | Resource-Initiated Allocation | - | - | + | + |
| 22 | Resource-Initiated Execution – Allocated Work Item | + | + | + | + |
| 23 | Resource-Initiated Execution – Offered Work Item | + | + | - | - |
| 24 | System-Determined Work List Management | + | + | + | o |
| 25 | Resource-Determined Work List Management | + | + | + | o |
| 26 | Selection Autonomy | + | + | + | + |
| 27 | Delegation | + | + | - | - |
| 28 | Escalation | + | + | - | - |
| 29 | Deallocation | - | - | - | - |
| 30 | Stateful Reallocation | +/- | + | - | - |
| 31 | Stateless Reallocation | - | - | - | - |
| 32 | Suspension/Resumption | +/- | +/- | - | - |
| 33 | Skip | - | - | + | o |
| 34 | Redo | - | - | + | o |
| 35 | Pre-Do | - | - | + | o |
| 36 | Commencement on Creation | - | - | - | - |
| 37 | Commencement on Allocation | - | - | - | - |
| 38 | Piled Execution | - | - | - | - |
| 39 | Chained Execution | - | - | + | - |
| 40 | Configurable Unallocated Work Item Visibility | - | - | - | o |
| 41 | Configurable Allocated Work Item Visibility | - | - | + | o |
| 42 | Simultaneous Execution | + | + | +/- | + |
| 43 | Additional Resources | - | - | - | - |

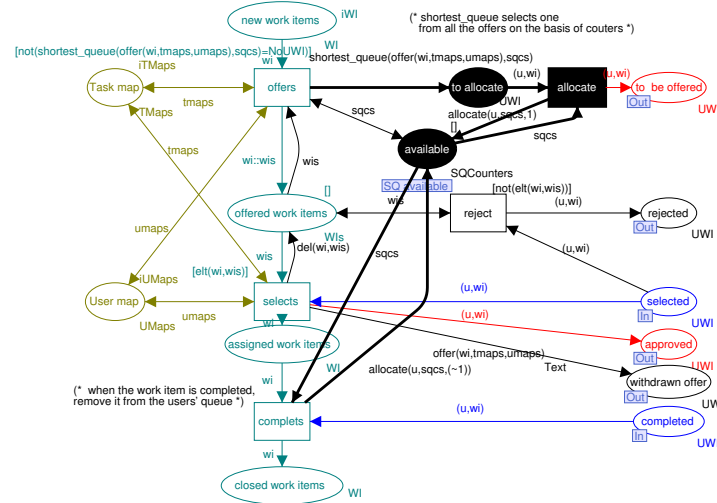


Fig. 4. Push Patterns - Shortest Queue

Table 5 shows an overview of the patterns. It also shows whether a pattern is directly supported by the three systems (SW = Staffware, FN = FileNet, FW = FLOWer) and the Basic Model (BM). We cannot elaborate on each of the patterns, but we will discuss one to illustrate our work. None of the systems supports *Pattern 17: R-SHQ (Shortest Queue)*. Pattern 17 is one of the push patterns, i.e., a pattern to push work to a specific user. For this pattern, a new work item is pushed to the user with the shortest queue of all users that qualify. This implies that each user has a counter to count the number of pending work items. Based on this counter, the work is distributed. As figure 4 shows, the required changes to the Basic Model are minimal. A counter is introduced for each user (token in place *available*) and function *shortest_queue* is used to select one user from the set of possible users based on these counters. Similarly, most of the other patterns can be realized quite easily.

Table 5 shows that the Basic Model supports less patterns than any of the three systems. This makes sense since each of the system-specific models can be seen as an extension of the Basic Model. It is interesting to see that existing systems typically support less than half of the patterns directly. This reveals typical limitations of contemporary products. Some of the patterns are considered out-of-scope for the reference model we are aiming at (marked with “o”). These are typically patterns directly depending on control-flow functionality, while we prefer to focus exclusively on work distribution. Each of the patterns not marked with “o” can easily be added to the Basic Model separately. However, the patterns tend to interact. For example, what does “Shortest Queue” (Pattern 17) mean if multiple resources work on the same item (Pattern 43)? Therefore, we are still looking for a suitable CPN model that captures many patterns while still being intuitive and relatively simple, i.e., a more comprehensive reference

model for work distribution. For this quest we want to use the results presented in this paper.

5 Related Work

Since the early nineties workflow technology has matured and several textbooks have been published, e.g., [1, 12, 17, 26, 28]. During this period many languages for modeling workflows have been proposed. These languages range from generic Petri-net-based languages to tailor-made domain-specific languages.

Despite the central role that resources play in workflow management systems, there is a surprisingly small body of research into resource and organizational modeling in a workflow context [24]. In their early work, Bussler and Jablonski [8] identified a number of shortcomings of workflow management systems when modeling organizational and policy issues. In subsequent work [17], they presented one of the first broad attempts to model the various perspectives of workflow management systems in an integrated manner including detailed consideration of the organizational/resource view.

One line of research into resource modeling and enactment in a workflow context has focused on the characterization of resource managers that can manage organizational resources and enforce resource policies. In [11], the design of a resource manager is presented for a workflow management system. It includes a high level resource model together with proposals for resource definition, query and policy languages. Similarly, in [25] an abstract resource model is presented in the context of a workflow management system although the focus is more on the efficient management of resources in a workflow context than the specific ways in which work is allocated to them. In [16], a proposal is presented for handling resource policies in a workflow context. Three types of policy – qualification, requirement and substitution – are described together with a means for efficiently implementing them when allocating resources to activities.

Another area of investigation has been into ensuring that only suitable and authorized users are selected to execute a given work item. The RBAC (Role-Based Access Control) model [14] presents an approach for doing this. Whilst effective, RBAC models tend to focus on security considerations and neglect work distribution aspects.

Several researchers have developed meta-models, i.e., object models describing the relation between workflow concepts, which include work allocation aspects, cf. [3, 27–29, 36]. However, these meta-models tend to focus on the structural description of resource properties and typically do not describe the dynamical aspects of work distribution.

The work reported in this paper can be seen as an extension of the *workflow patterns initiative* (cf. www.workflowpatterns.com). Besides a variety of control-flow [2] and data [38] patterns, 43 resource patterns [37, 39] have been defined. This paper complements the work of resource patterns [37, 39] by providing executable models for work distribution mechanisms.

6 Conclusions

This paper is a first step towards a comprehensive reference model for work distribution in process-aware information systems (i.e., workflow management systems and beyond). To assist in understanding work distribution better, we used the CPN language and CPN Tools to model and analyze different mechanisms. To serve as a reference, we provided a basic model that can be seen as the “greatest common denominator” of existing workflow management systems. This model was extended and specialized for three specific systems (Staffware, FileNet, and FLOWer). The basic model already captures many of the so-called *resource patterns* defined earlier. However, we also modeled more advanced patterns by extending the basic model. In contrast to existing research mainly using static models (e.g., UML class diagrams), we focused on the dynamics of work distribution.

Our experiences revealed that it is relatively easy to model and analyze the systems and patterns using CPN Tools. This suggests that CPN language and the basic CPN model are a good basis for future research. We plan to extend the Basic Model into a more *comprehensive reference model for work distribution*. First, we want the model to be able to capture the typical functionality offered by existing systems. One can think of this as the “Least Common Multiple” of existing functionality. The corresponding CPN model will be much more complicated than the Basic Model used now. However, it can serve as a reference for organizations that want to implement more advanced functionality. The goal is to design and implement distribution mechanisms that overcome the limitations of existing systems. An important ingredient will be to use insights from socio-technical design [7, 10, 13, 43] as mentioned in the introduction.

References

1. W.M.P. van der Aalst and K.M. van Hee. *Workflow Management: Models, Methods, and Systems*. MIT press, Cambridge, MA, 2002.
2. W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros. Workflow Patterns. *Distributed and Parallel Databases*, 14(1):5–51, 2003.
3. W.M.P. van der Aalst and A. Kumar. Team-Enabled Workflow Management Systems. *Data and Knowledge Engineering*, 38(3):335–363, 2001.
4. W.M.P. van der Aalst, M. Weske, and D. Grünbauer. Case Handling: A New Paradigm for Business Process Support. *Data and Knowledge Engineering*, 53(2):129–162, 2005.
5. J. Becker, M. Kugeler, and M. Rosemann, editors. *Process Management: A Guide for the Design of Business Processes*. Springer-Verlag, Berlin, 2003.
6. P. Bernus. Generalised Enterprise Reference Architecture and Methodology, Version 1.6.3. IFIP/FAC Task Force on Architectures for Enterprise Integration, March 1999.
7. J. Bowers, G. Button, and W. Sharrock. Workflow From Within and Without: Technology and Cooperative Work on the Print Industry Shopfloor. In *The Fourth European Conference on Computer-Supported Cooperative Work (ECSCW 95)*, pages 51–66, Stockholm, September 1995. Kluwer Academic Publishers, Dordrecht, The Netherlands.

8. C. Bussler and S. Jablonski. Policy Resolution for Workflow Management Systems. In *Proceedings of the 28th Hawaii International Conference on System Sciences*, page 831. IEEE Computer Society, 1995.
9. T. Curran and G. Keller. *SAP R/3 Business Blueprint: Understanding the Business Process Reference Model*. Upper Saddle River, 1997.
10. L. U. de Sitter, J. F. den Hertog, and B. Dankbaar. From complex organisations with simple jobs to simple organizations with complex jobs. *Human Relations*, 510(5):497–534, 1997.
11. W. Du and M.C. Shan. Enterprise Workflow Resource Management. In *Ninth International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises (RIDE-VE'99)*, pages 108–115, Sydney, Australia, 1999. IEEE Computer Society Press.
12. M. Dumas, W.M.P. van der Aalst, and A.H.M. ter Hofstede. *Process-Aware Information Systems*. Wiley & Sons, 2005.
13. F.M. van Eijnatten and A.H. van der Zwaan. The Dutch IOR approach to organisation design. An alternative to business process re-engineering? *Human Relations*, 51(3):289–318, 1998.
14. D.F. Ferraiolo, R. Sandhu, S. Gavrila, D.R. Kuhn, and R. Chandramouli. Proposed NIST Standard for Role-Based Access Control. *ACM Transactions on Information and System Security*, 4(3):224–274, 2001.
15. FileNET. *FileNet Business Process Manager 3.0*. FileNET Corporation, Costa Mesa, CA, USA, June 2004.
16. Y.N. Huang and M.C. Shan. Policies in a Resource Manager of Workflow Systems: Modeling, Enforcement and Management. Technical Report HP Tech. Report, HPL-98-156, Palo Alto, CA, USA, 1999. Accessed at <http://www.hpl.hp.com/techreports/98/HPL-98-156.pdf> on 20 March 2005.
17. S. Jablonski and C. Bussler. *Workflow Management: Modeling Concepts, Architecture, and Implementation*. International Thomson Computer Press, London, UK, 1996.
18. K. Jensen. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use. Volume 1*. EATCS monographs on Theoretical Computer Science. Springer-Verlag, Berlin, 1997.
19. K. Jensen and G. Rozenberg, editors. *High-level Petri Nets: Theory and Application*. Springer-Verlag, Berlin, 1991.
20. G. Keller, M. Nüttgens, and A.W. Scheer. Semantische Prozessmodellierung auf der Grundlage Ereignisgesteuerter Prozessketten (EPK). Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 89 (in German), University of Saarland, Saarbrücken, 1992.
21. G. Keller and T. Teufel. *SAP R/3 Process Oriented Implementation*. Addison-Wesley, Reading MA, 1998.
22. E. Kindler. On the Semantics of EPCs: A Framework for Resolving the Vicious Circle. In J. Desel, B. Pernici, and M. Weske, editors, *International Conference on Business Process Management (BPM 2004)*, volume 3080 of *Lecture Notes in Computer Science*, pages 82–97. Springer-Verlag, Berlin, 2004.
23. L.M. Kristensen, S. Christensen, and K. Jensen. The Practitioner's Guide to Coloured Petri Nets. *International Journal on Software Tools for Technology Transfer*, 2(2):98–132, 1998.
24. A. Kumar, W.M.P. van der Aalst, and H.M.W. Verbeek. Dynamic Work Distribution in Workflow Management Systems: How to Balance Quality and Performance? *Journal of Management Information Systems*, 18(3):157–193, 2002.

25. B.S. Lerner, A.G. Ninan, L.J. Osterweil, and R.M. Podorozhny. Modeling and Managing Resource Utilization in Process, Workflow, and Activity Coordination. Technical Report UM-CS-2000-058, Department of Computer Science, University of Massachusetts, August 2000. Accessed at <http://laser.cs.umass.edu/publications/?category=PROC> on 20 March 2005.
26. F. Leymann and D. Roller. *Production Workflow: Concepts and Techniques*. Prentice-Hall PTR, Upper Saddle River, New Jersey, USA, 1999.
27. M. Zur Muehlen. Evaluation of Workflow management Systems Using Meta Models. In *Proceedings of the 32nd Hawaii International Conference on System Sciences - HICSS'99*, pages 1–11, 1999.
28. M. Zur Muehlen. *Workflow-based Process Controlling: Foundation, Design and Application of workflow-driven Process Information Systems*. Logos, Berlin, 2004.
29. M. zur Muhlen. Organizational Management in Workflow Applications Issues and Perspectives. *Information Technology and Management*, 5(3–4):271–291, July–October 2004.
30. Pallas Athena. *Flower User Manual*. Pallas Athena BV, Apeldoorn, The Netherlands, 2002.
31. Pallas Athena. *Protos User Manual*. Pallas Athena BV, Plasmolen, The Netherlands, 2004.
32. M. Pesic and W.M.P. van der Aalst. Modeling Work Distribution Mechanisms using Colored Petri Nets. BETA Working Paper Series, Eindhoven University of Technology, Eindhoven, 2005.
33. W. Reisig and G. Rozenberg, editors. *Lectures on Petri Nets I: Basic Models*, volume 1491 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1998.
34. M. Rosemann. Application Reference Models and Building Blocks for Management and Control (ERP Systems). In P. Bernus, L. Nemes, and G. Schmidt, editors, *Handbook on Enterprise Architecture*, pages 596–616. Springer-Verlag, Berlin, 2003.
35. M. Rosemann and W.M.P. van der Aalst. A Configurable Reference Modelling Language. *Information Systems*, pages (accepted for publication, can be obtained via BPMcenter.org), 2005.
36. M. Rosemann and M. Zur Muehlen. Evaluation of Workflow Management Systems - a Meta Model Approach. *Australian Journal of Information Systems*, 6(1):103–116, 1998.
37. N. Russell, W.M.P. van der Aalst, A.H.M. ter Hofstede, and D. Edmond. Workflow Resource Patterns: Identification, Representation and Tool Support. In O. Pastor and J. Falcao e Cunha, editors, *Proceedings of the 17th Conference on Advanced Information Systems Engineering (CAiSE'05)*, volume 3520 of *Lecture Notes in Computer Science*, pages 216–232. Springer-Verlag, Berlin, 2005.
38. N. Russell, A.H.M. ter Hofstede, D. Edmond, and W.M.P. van der Aalst. Workflow Data Patterns. QUT Technical report, FIT-TR-2004-01, Queensland University of Technology, Brisbane, 2004.
39. N. Russell, A.H.M. ter Hofstede, D. Edmond, and W.M.P. van der Aalst. Workflow Resource Patterns. BETA Working Paper Series, WP 127, Eindhoven University of Technology, Eindhoven, 2004.
40. A.W. Scheer. *Business Process Engineering, Reference Models for Industrial Enterprises*. Springer-Verlag, Berlin, 1994.
41. A.W. Scheer. *ARIS: Business Process Modelling*. Springer-Verlag, Berlin, 2000.
42. Staffware. *Using the Staffware Process Client*. Staffware, plc, Berkshire, United Kingdom, May 2002.
43. F. M. van Eijnatten. *The Paradigm that Changed the Work Place*. Van Gorcum, Assen, The Netherlands, 1993.

An Open and Formalism Independent Meta-Model for Business Processes

Björn Axenath, Ekkart Kindler, Vladimir Rubin

Software Engineering Group, University of Paderborn,
Warburger Str. 100, D-33098 Paderborn, Germany
[axenath|kindler|vroubine]@uni-paderborn.de

Abstract. Business process reference models help adapting processes of an enterprise to the processes supported by standard software or help customising processes in standard software to fit the needs of some enterprise. The problem, however, is that, in many cases, the processes of the enterprise and the reference processes are modelled in different notations and formalisms, which makes the adaptation and customisation a difficult task. In this paper, we formalise the concepts of business process models and their relation as a meta-model in UML notation. Though this meta-model resembles existing architectures, it enjoys some important additional properties: It is independent of particular modelling formalisms, it is open so that new concepts for some aspect can be easily added, and it is operational in the sense that a workflow management system can be implemented based on the meta-model without even knowing the particular modelling formalisms. This way, the meta-model proposed here identifies the essential concepts of different modelling formalisms and, therefore, helps relating reference process models and process models even though they are modelled in different formalisms.

1 Introduction

Business process reference models help adapting processes of an enterprise to the processes supported by standard software, or they help customising processes of standard software to fit the needs of some enterprise. The problem, however, is that, in many cases, the processes of the enterprise and the reference processes are modelled in different notations and formalisms, which makes the adaptation and the customisation a difficult task. In order to deal with this problem, it is important to identify the common aspects and underlying concepts of business process models and business process reference models independently of a particular modelling formalisms. To this end, we formalize the concepts for modelling business processes as a meta-model. We use UML notation for formalizing this meta-model, since this allows us not only to present the meta-model, but, ultimately, to implement these concepts.

There is much work on ontologies for business process models, and there are various kinds of taxonomies, glossaries, and meta-models in the area of business process modelling (see Sect. 5). All these approaches help better understanding

the concepts of business process modelling. But, they exhibit one of the following two problems: Either they are focused on one particular formalism or notation for modelling business processes and are restricted to some particular aspects of business processes; or they are so general that mapping concrete formalisms to them is difficult and remains vague. Our meta-model reconciles the generality of its concepts and the independence from particular formalisms with a concise way of mapping concrete formalisms to these concepts. Technically, this is achieved by interfaces for integrating formalisms to this meta-model. This way, we distill and better understand the concepts involved in business process modelling – independently of a particular formalism. This understanding can help relating and comparing process models in different notations, it can help defining an interchange format for business process models, and it will eventually help switching from one formalism to another with respect to one aspect without changing the formalisms and models of other aspects of the process. Actually, our long-term goal is a formalism independent workflow engine.

In this paper, we will present and formalize the basic ideas of our meta-model for business process modelling. The principles governing the design of this meta-model were the following: 1. It should cover all basic aspects of business process models and should not be biased towards or focused on one of these aspects. 2. It should be open so that other aspects can be easily added and integrated to it. 3. In particular, there should be clear interfaces for the different aspects and a mechanism for their integration. 4. It should be independent of a particular notation for business process models, which allow us to map existing business process modelling notations to this meta-model. 5. It should be compatible with existing architectures of workflow management systems (e.g. [7]).

In this paper, the meta-model will be defined on different levels of abstraction and by different techniques [14]. First of all, we briefly explain and define all relevant concepts and introduce a *controlled vocabulary* for the domain of business process models. In a second step, we provide meta-models for these concepts in the Meta Object Facility (MOF [13]) in UML notation. The main idea is that the different aspects of a business process can be modelled independently of each other and can be easily combined without knowing the underlying formalisms. To this end, the meta-models for the different aspects define interfaces that can be implemented by different concrete formalisms. Note that, for lack of space, we will not deal with the details of the information aspect of business processes here. The details are presented and discussed in a technical report [1] on *AMFIBIA: A Meta-model For the Integration of Business process modelling Aspects*.

2 Concepts and Terminology

In this section, we introduce the basic concepts and terminology used in business process modelling and define a controlled vocabulary for the domain of business processes. In the literature, there are many proposals using different terms, which are not consistent with each other and no terminology is fully accepted. Our definitions and terminology has been strongly influenced by [7, 17, 12].

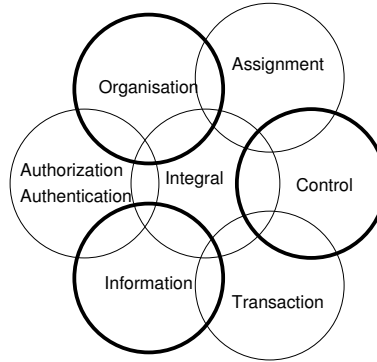


Fig. 1. Aspects of business processes

2.1 Overview

A *business process* consists of a set of *activities* that are executed in some enterprise or administration according to some rules in order to achieve certain goals. A *business process model* is a more or less formal and more or less detailed description of the persons and artefacts involved in the execution of a particular business process and of the rules governing their execution. An *instance*, i. e. a particular execution of a business process model, is often also called a business process. In order to avoid confusing instances and models, we call an instance of a business process model a *case*.

It is now well-accepted that there are different aspects of business processes that can be modelled and investigated independently of each other: The *control aspect* basically describes the order in which the different activities are executed. The *organizational aspect* describes the organization structure and, in particular, the resources and agents, and in which way they are involved in the activities of the business process. The *informational aspect* describes the information that is involved in a business process, how it is represented, and how it is propagated among different activities. The details of these aspects and the concepts modelled in a particular aspect will be discussed below. Actually, there are many more aspects, some of which are shown in Fig. 1 (see [1] for details).

In many papers, the control aspect is considered to be the most salient aspect of business processes, and in many modelling formalisms and notations, this aspect is used for integrating all other aspects. In our proposal, we do not use the control aspect for the integration of the other aspects. Rather, we single out the concepts that are common to all aspects. We call these concepts the *integral part* of business processes, which comprises basically the *activities* of the process. An activity itself is an instance of some particular *task*; only when a particular case is executed the tasks will be instantiated to an activity.

A *task* comprises pieces of work that conceptually belong to each other. A task can either be *atomic* or *compound*. An atomic task is not split into further parts on a given level of abstraction. Dependent on the purpose of the model, an

atomic task can be associated with a procedure or an application that supports the execution of this task; this, however, is subject to a special aspect already: the *application aspect*. A compound task refers to a *subprocess*, which defines the details of this task; this can be defined by another business process. Actually, the distinction of atomic and compound tasks belongs to the *structuring aspect* already; but, we do not introduce this aspect here.

2.2 Control Aspect

The control aspect defines the order in which the tasks of a business process are instantiated and in which the corresponding activities are executed. Note that the order of the execution does not need to be sequential; it can be a partial order representing the causal dependencies among the activities.

For defining this order, the formalism refers to the tasks defined in the integral part of the business process. There are many different ways, formalisms, and notations for defining the order in which the tasks, resp. the corresponding activities must be executed. In order to be universal, we do not fix a particular formalism for defining the control of a business process. We assume only that there is a concept of a *state*. In a given state, it must be clear which tasks are enabled (i. e. which tasks can be instantiated to activities), how the instantiation of a task to an activity changes the state, and how the termination of this activity changes the state. We will discuss this in more detail later in Sect. 4.2.

2.3 Informational Aspect

The informational aspect of a business process model defines the information involved in a business process as well as the propagation of information among different activities. All information involved in a business process can be considered to be *documents*, where a document is an artefact representing some piece of information. The information aspect of a business process basically defines the structure of the involved documents and their relation. Moreover, it defines how documents are propagated between tasks.

Similar to processes and cases and to tasks and activities, we must distinguish between *document models* and *document instances* (documents for short), where a document model defines the structure of a document instance. The *information model* comprises all document models as well as the relation among the different documents. As for tasks, a document can be *atomic* or *compound*. An atomic document is an unstructured text or piece of data (resp. the structure is not represented in the model), whereas a *compound document* is structured and consists of two or more *sub-documents*.

2.4 Organizational Aspect

The organizational aspect of a business process model defines the structure of the organisation in which the business process is executed, i.e. its *organisational*

units and the *relations* among them; and it defines the *resources* and *agents* within these organizational units, where we use the term agents in order to refer to human resources. Moreover, the organizational aspect of a business process model defines, which resources and agents could possibly execute a particular task resp. activity; these are called the possible *assignments* for that task. In a business process model each task will be equipped with a *resource descriptor*, which defines the possible assignments once the task becomes enabled.

The structure of the organization is modelled in the *organization model*. Note that the organisation model captures only the static part of the organization such as departments and groups. It does not deal with the dynamic parts of concrete agents and resources. Therefore, the possible assignments of agents and resources to some task are defined by a *resource descriptor*, possibly via their *positions*, *roles* or via *relations* (such as substitute) among resources.

When it comes to the execution of a business process model in a workflow management system, the concrete resources and agents, their positions and roles will be maintained by some administrator, such that the workflow management system can assign tasks to the concrete agents.

2.5 Static and Dynamic Concepts

In the presentation of the concepts of business processes, we have dealt with two kinds of concepts. The first kind, were the concepts occurring in the business process model itself, e.g. its tasks, the document models, and the roles etc. The second kind are the instances of the first ones such as cases, activities, document (instances), and resources. Only the concepts of the first category do occur in the models, the concepts of the second category are necessary only for the definition of the meaning of a model in terms of its instances, which, basically, defines the dynamic behaviour of a model. In the following discussions and, in particular, in the meta-model, we will carefully distinguish between these categories. We call the first kind of concepts *static concepts* or *modelling concepts*, and we call the second kind *dynamic concepts* or *instance concepts*.

3 Meta-Modelling Techniques

Up to now, we have presented the basic idea and concepts of our business process modelling meta-model. In Sect. 4, we will present all the details necessary for a future implementation of this meta-model. It is defined with the help of MOF and UML techniques. These techniques are explained in this section with the help of some examples from the control aspect of a business process.

3.1 Models and Meta-Models

In this section we discuss the concepts of models, meta-models and instances. A *model* is an abstraction of the real world. In Fig. 2, we present a concrete workflow net as an example of a model for the control aspect of a business

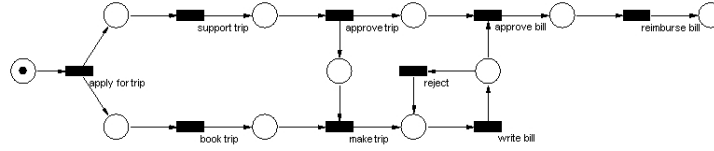


Fig. 2. Example of a model

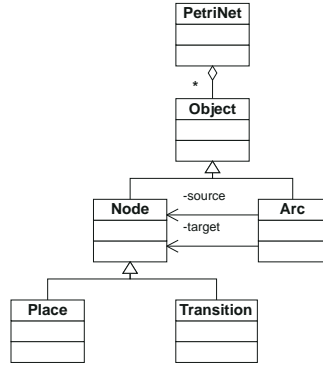


Fig. 3. PNML Core Model

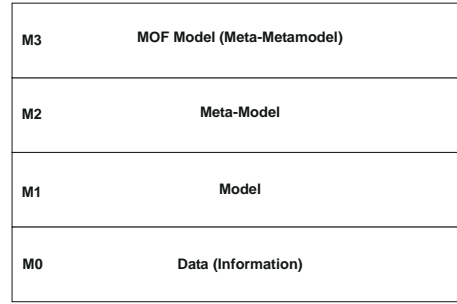


Fig. 4. MOF-compliant ontology

process. The example is a “Conference Trip”, different aspects of which will be added in Sec. 4.

We need different formalisms for defining our models. We use Petri Nets for the control aspect, organization charts for the organizational aspect, and ER-diagrams for the informational aspect. In order to use the different formalisms, we need to define the concepts and notations of these particular formalisms. This can be done by providing a *meta-model* for the formalism; i.e. the meta-model defines the language or notation for expressing the model. For example, for computer programs the grammar defining their syntax is their meta-model; for XML documents an XML-Schema or a DTD is a meta-model. For our workflow net example, the Petri Net Markup Language (PNML) Core Model [11] is the meta-model (see Fig. 3). A place of the workflow net is an instance of the class *Place* of the PNML Core Model. The meta-model defines not only classes, but also relations between them. Therefore, UML class diagrams are an appropriate technique for defining such kinds of meta-models.

The meta-modelling technique can not only be used for defining the constructs and rules for building models, it can be used also for defining the concepts of a domain of interest and their relation. A combination of such a meta-model and a controlled vocabulary defines the ontology of the domain.

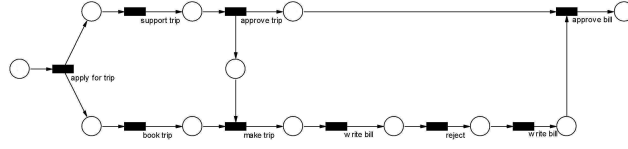


Fig. 5. Example of an instance

Of course, the notation for defining the meta-models must also be defined in some way. To this end, we use the techniques of UML and the Meta Object Facility (MOF) [13]. MOF is a framework for defining models and meta-models. Basically it consists of four layers (see Fig. 4): the MOF model (M3), a meta-model (M2), a model (M1), and an information layer (M0).

In our example, the meta-model layer contains the PNML Core Model, the model layer contains the workflow net, and the information layer contains the instances of the workflow net. Figure 5 shows one example of an instance of a workflow net, which is an occurrence net. This occurrence net documents one possible real execution of the process. It shows the current state of the process and its history. In our example, we can see that, in this process instance, the bill was rejected once, but later the bill was approved – but not yet fully completed.

Altogether, our meta-models is defined on the M2 layer and the business process models will be on the M1 layer of MOF.

3.2 Mapping the Formalisms

The concepts of business process modelling as discussed in Sect. 2 are independent of concrete modelling formalisms such as Petri nets, organization charts or ER-diagrams. In order to use our meta-model with concrete formalisms, we need a mechanism for integrating these formalisms to the meta-model. To this end, we assume that there is a meta-model for the formalism. This meta-model must be mapped to our meta-model. Technically, the classes of our meta-model are interfaces, and the classes of the meta-model of the formalism implement these interfaces. Figure 6 gives an overview of this mechanism. The interfaces of each aspect define the *formalism independent* meta-model, and the meta-model of a formalism implementing such an interface define a *formalism dependent* meta-model. In Sect. 4.2, we will show how the PNML Core Model implements the formalism independent meta-model of the control aspect.

4 Business Process Integration Meta-Model

In this section, we introduce the meta-models for two basic aspects of business processes and we show how concrete formalisms implement our formalism independent meta-models. The informational aspect is explained in [1].

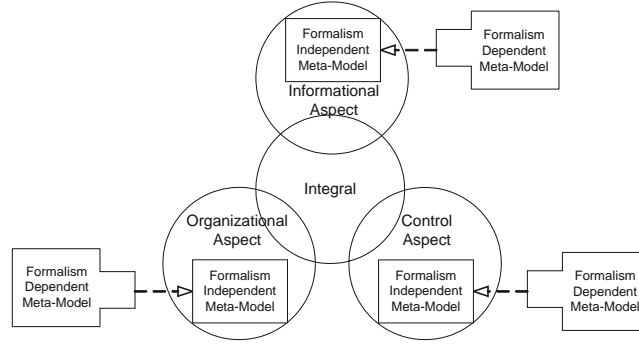


Fig. 6. Mapping of the formalism dependent meta-models

4.1 A Conference Trip Example

Before going to the meta-model level, we discuss two aspects of a concrete business process. As an example we have chosen a conference trip process, where the models of the aspects are independent of each other and use different formalisms.

The Control Aspect of the Example In Fig. 2, we have seen an example of a model for the control aspect of a business process already. The formalism is a special version of Petri nets called *workflow nets* [17]. It defines the different tasks of a conference trip and the order in which they are executed. The tasks are the transitions (represented as rectangles) of the Petri net. A Petri net defines this behaviour in terms of a *marking*, which is a number of tokens on its places (represented as black dots in the circles). Initially, there is only one token on the initial place *in*. At this stage, only transition “apply for trip” is enabled. After starting and finishing the corresponding task, the transitions “support trip”, which corresponds to the task of a superior countersigning the trip application, and the transition “book trip” are enabled. This means that these two tasks could be executed concurrently. Note that the actual trip may be made only if the trip application has been approved before. After the trip, the employee may apply for reimbursement of the travel expenses. Note that, at this stage, there might be an iteration: If the bills for the trip are rejected, the billing activity will be repeated. The process is finished, when a token arrives at place *out*.

The Organizational Aspect of the Example The “Conference Trip” process can possibly take place in some company, which has the structure shown in Fig. 7. The structure of the company is presented in a matrix form. The organization “SWCompany” is an organizational unit, which contains four other organizational units. Each organizational unit contains organizational positions. Each organizational position implies a role. For example, the CRM unit contains a position such as “Salesman”. An agent occupies an organizational position and

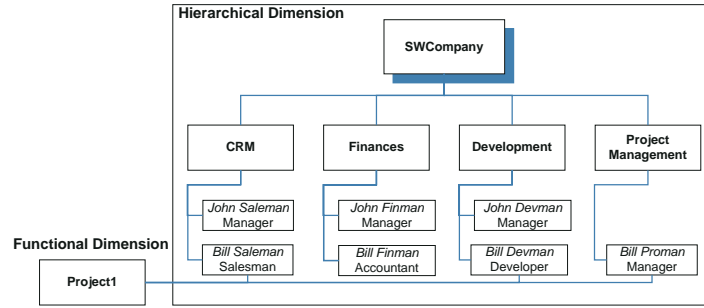


Fig. 7. Example of matrix organizational structure

thus is assigned a role. For example, “Bill Saleman” is a salesman. Altogether, such an organizational tree represents the hierarchical dimension of the matrix structure of the organization.

The other dimension is the functional one. It is based on the projects in which the members of different organizational units are involved. For example, the salesman from the CRM unit, the developer from the Development unit and the manager from the Project Management unit participate in the same project “Project1”. Thus, the vertical structure of “SWCompany” shows the hierarchical dimension and the horizontal – the functional one.

For the given organizational structure, we can give the responsibilities schema in the context of the “Conference Trip” process, which basically has to answer the question “Who can execute the task?”. An example for such a schema is presented in the technical report [1].

4.2 The Meta-Model

In this section, we present the meta-model for business process models, with a clear separation of the integral part and the meta-models for the different aspects. Technically, the meta-models for the different aspects are interfaces that must be implemented by a particular formalism in order to implement it as a model for this particular aspect.

In the following, we first give a meta-model for the integral parts of business process models. Then, we give the meta-models for the concepts of the different aspects. For each aspect, we show how this aspect can be implemented by a particular formalism.

Integral Part Figure 8 shows the meta-model of the integral part of business process models, which are independent of a particular aspect. The left-hand side shows the static concepts and the right-hand side shows the dynamic concepts.

A *business process model* (BPM) consists of a set of *tasks*. A task can either be a *basic* (or an *atomic*) task, or it can be a *compound task*; a compound task

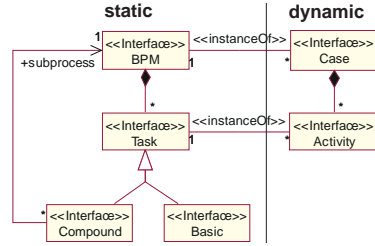


Fig. 8. The integral part

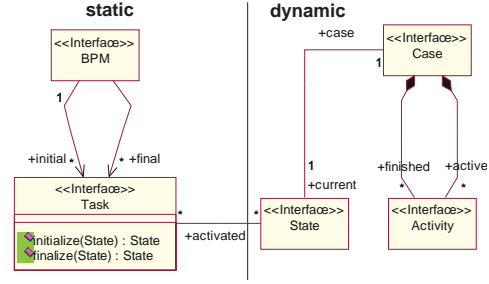


Fig. 9. The control aspect

refers to another business process, a *subprocess*, which defines this particular task. A *case* is an instance of a particular business process. While running, different tasks will be instantiated in a case, which are called the *activities*.

Control Aspect In this section, we present the meta-model for the control aspect of business process models. To this end, we refer to the concepts of the integral part (see Fig. 8) and extend them by additional features. These are shown in Fig. 9.

For the control aspect, a process model is equipped with *initial* and *final* tasks. The initial tasks define those tasks that initiate a new case. The final tasks identify those tasks that terminate the execution of the resp. case¹

In order to define the control model of a process, there is the concept of a *state*, which actually sits in the middle between the static and the dynamic concepts of business processes. Each case has a current state, which in turn defines the tasks that could possibly be started in the current state; these are called the *activated* tasks. Each task defines, how the initialisation of this task changes the state and how the finalisation of the corresponding activity changes the state of the case. Moreover, a case consists of a set of activities that are active at a particular moment, and it consists of activities that are finished already.

Thus far, the meta-model for the control aspect is independent of a particular formalism for modelling the behaviour. It requires only that there is a concept of a state and state changes. This can be implemented by different formalisms. Here, we show how Petri nets can be used for implementing the control aspect. Figure 10 shows the meta-model of Petri nets (PNML) implementing the concepts of the control aspect of business processes. It consists of transitions and places. A marking consists of a multiset of places. The transitions implement the tasks, the markings implement states, and the firing rule of Petri nets implements the state changes. The method *initialize* removes one token from each

¹ For experts in UML modelling, we mention that, actually, there are separate interfaces for a BPM, a Task and a Case for every aspect, which is a technique known from aspect oriented modelling. But, we do not go into the details here.

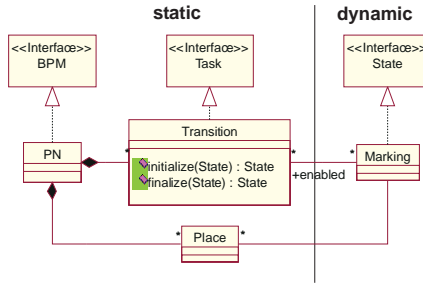


Fig. 10. A Petri net implementation

of its input places, and the method `finalize` adds a token to each of its output places. The enabledness of a transition in a particular marking implements the set of activated tasks.

Here, we used Petri nets for implementing a formalism for the control aspect of business process models. But, it is easy to see that any other formalism that has a semantics based on states and state changes (transitions) can be used for implementing the control aspect; which virtually applies to all formalisms used in control models of business processes, though some may be more complicated.

Organizational Aspect In this section, we present a typical meta-model for the organizational aspect of business process models. Here, we refer to the integral part and show its relations to the organizational aspect (see Fig. 11).

In order to define which resources can execute a task, we introduce the *ResourceDescriptor*. The *ResourceDescriptor* defines the set of *possible assignments*, in some *context* of the case. The context of a case, basically, shows the assignments that were done to some earlier activities. The assignment contains a set of resources which could execute the activity. This set of classes provides basically the ontology of the organizational aspect, which must be implemented by a concrete formalism.

Figure 12 shows the concepts that are supported by virtually all formalism for the organizational modelling. This meta-model can be divided into three parts:

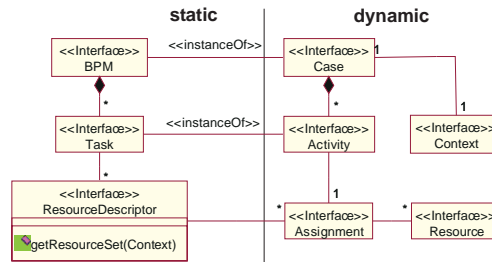


Fig. 11. The Organizational aspect meta-model

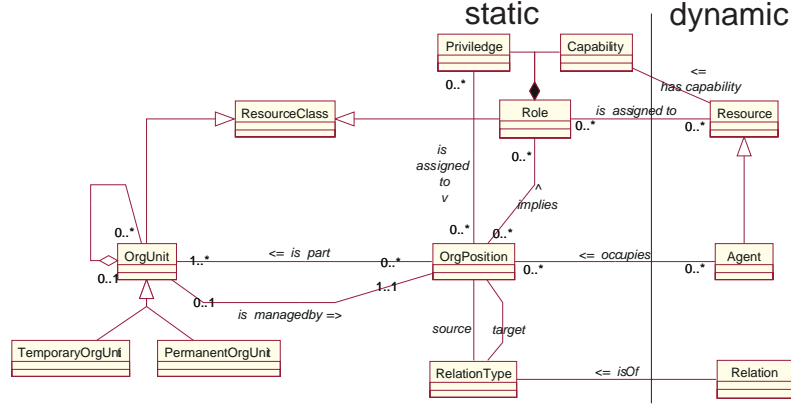


Fig. 12. The organizational structures meta-model of business processes

organizational structure, functional structure, and resources. The organizational and the functional structure classes serve as a classification of resources. The organizational structure consists of the *OrgUnit* (Organizational Unit) and its subclasses and the *OrgPosition* (Organizational Position). The Organizational Unit is a group of people working together and organized for some purpose. Organizational Unit can be either *Temporal* or *Permanent*. The Organizational Unit is formed from Organizational Positions. Thus, the Organizational Position can be considered as an atomic Organizational Unit. The functional structure consists of the *Role*, the *Privilege* and the *Capability*. As it is shown in the meta-model diagram, an Organizational Position can imply some Roles. The Role is actually a group of resources exhibiting a set of specific skills and/or qualifications. The Role is composed of Privileges and Capabilities. The Privilege is usually assigned to the appropriate Position. The Capability is a direct property of a *Resource*. The Resource is a person, machine or application, which can be assigned a task. The Resource is requested to perform an activity at runtime. The *Agent* is a human Resource. There can be different *Relations* between Resources within a particular process instance, for example, a substitution relation. Hence, there must exist also *Relation Types*, which define different kinds of Relations in an Organizational.

The meta-model consists of static and dynamic parts. Resources, Agents and Relations belong to the dynamic part and the others to the static.

Next, we show how the meta-model from Fig. 12 can be mapped to the organizational aspect (see Fig. 13). The ResourceDescriptor basically consists of three parts: a Role, an Organizational Unit and some additional constraints. The *SpecificRD* (Specific ResourceDescriptor) implements the ResourceDescriptor and, thus, also provides a set of possible assignments, which are now calculated from the the Role, the OrgUnit and the Restriction. The Role and the OrgUnit interfaces are the interfaces from the organizational structures meta-model; they specify the functional and the organizational classification of the

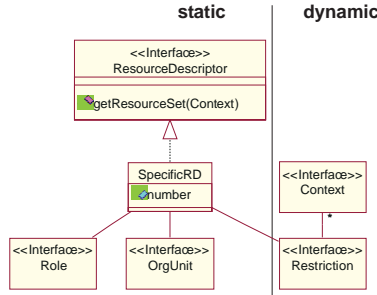


Fig. 13. The organizational aspect with structure

resource who could execute a task. The Restriction interface specifies the properties of the desired resource based on the context of the case, i.e. based on the assignments that were already done in the case.

It is easy to see that any formalism for organizational models can directly implement the ResourceDescriptor interface. In the mapping above, we implemented the ResourceDescriptor in a more structured way by using interfaces for Roles, Organizational Units and Restrictions. Since almost every formalism defines these concepts, it is not necessary to implement the ResourceDescriptor directly, the formalism can be mapped to these interfaces; then SpecificRD implements a ResourceDescriptor. We recommend this style of implementing a ResourceDescriptors because it identifies the organizational aspect details clearer and prescribes the correct usage of the organizational structures classes.

4.3 Integration

In the previous sections, we presented the meta-models for the integral part of business process models and the meta-models resp. interfaces for two different aspects of business process models. We have shown how a particular meta-model for some formalisms can be mapped to the interfaces of a particular aspect. In order to validate the concept, we show that a business process can be fully defined in terms of the concepts of the formalism independent meta-model.

Actually, we could implement a formalism independent workflow engine based on these interfaces. Though we did not implement such a formalism independent workflow engine yet, we have shown in a ‘Gedankenexperiment’ that a workflow engine based only on the interfaces of the different aspects will work [1].

5 Related Work

In this section, we give a brief overview of related work. For lack of space, we first discuss the work from the area of reference meta-models, where we restrict ourselves to the meta-models presented in a more or less formal way. Second, we

refer to the research in the area of reference models, where we present the most general work only.

Wil van der Aalst and Kees van Hee [17] present a reference framework for defining business processes with a focus on workflow management systems. Leymann and Roller [12] discuss the basics of the workflow technology and its aspects, the models and meta-models of business processes, and the workflow management systems. Their book presents the definitions of the meta-model using precise syntax and semantics. The Workflow Reference Model of the WfMC [7] provides a common vocabulary for describing a business process and its aspects. A discussion paper of the WfMC about the common object model [8] is a proposal for using the object-oriented technology in this area. In the XML Process Definition Language Specification of the WfMC [21], the meta-model of the process definition, containing the main entities, their relationships and attributes, was defined. The book of Jablonski et al. [9] discusses the topic of the meta-modelling presenting the schema of the meta-levels and describing relations between them. In [3] the motivation and concepts of using the Enterprise Process Modelling Language (EPML), which considers different aspects of business processes, are presented.

The book “Referenzmodellierung” [2] covers the general concepts, techniques and applications of reference models. Fettke and Loos present the background ideas in the research field of reference models and applications to such domains as E-Business and business engineering in their works [5, 6]. Scheer discusses the application of reference models to the industrial business processes in the context of the Architecture of Integrated Information Systems (ARIS) [16].

The goal of our research lies in combining the different reference meta-models and reference models.

6 Conclusion

In this paper, we presented an ontology for business process modelling, by first introducing a controlled vocabulary and then formalizing it as a meta-model, which we call *AMFIBIA*. As discussed in Sect. 5, the idea of devising an ontology or a meta-model for business process models is not new at all. The justification for yet another one is its new focus: Our meta-model shows that the different aspects of business processes can be modelled independently of each other and that it is possible to integrate them via the integral part. It is open for additional aspects and is not biased towards any aspect. The meta-model is independent of any particular modelling formalism and, actually, defines an interface that can be implemented by most formalisms for that particular aspect.

Though, the presented meta-model is not restricted to business process reference models, it helps in this area by identifying the important aspects and concepts independently of a particular formalism. And, once implemented, it provides a technological basis for using reference models with different formalisms.

References

1. B. Axenath, E. Kindler, and V. Rubin. The aspects of business processes: An open and formalism independent ontology. Technical Report TR-RE-05-256, University of Paderborn, <http://wwwcs.uni-paderborn.de/cs/kindler/Publikationen/copies/AKR05.pdf>, April 2005.
2. J. Becker and P. Delfmann, editors. *Referenzmodellierung*. Physica-Verlag, 2004.
3. N. P. Dalal, M. Kamath, W. J. Kolarik, and E. Sivaraman. Toward an integrated framework for modeling enterprise processes. *Commun. ACM*, 47(3):83–87, 2004.
4. J. Evermann and Y. Wand. Towards Ontologically Based Semantics for UML Constructs. In *Proceedings of the 20th International Conference on Conceptual Modeling*, pp. 354–367. Springer 2001.
5. P. Fettke and P. Loos. Referenzmodellierungsforschung. *WIRTSCHAFTSINFORMATIK*, 46(5):331–340, 2004.
6. P. Fettke and P. Loos. Der Beitrag der Referenzmodellierung zum Business Engineering. *HMD - Praxis der Wirtschaftsinformatik*, 241:18–26, 2005.
7. D. Hollingsworth. The Workflow Reference Model. Technical Report TC00-1003, The Workflow Management Coalition (WfMC), January 1995.
8. D. Hollingsworth. A Common Object Model Discussion Paper. Technical Report WfMC-TC-1023, WfMC, March 1999.
9. S. Jablonski, M. Böhm, and W. Schulze. *Workflow-Management Entwicklung von Anwendungen und Systemen*. dpunkt.verlag, 1997.
10. D. Karagiannis and H. Kühn. Metamodeling platforms. In K. Bauknecht, A. Min Tjoa, and G. Quirchmayer, editors, *Proceedings of the Third International Conference EC-Web, LNCS 2455*, page 182. Springer-Verlag, September 2002.
11. E. Kindler. Using the Petri Net Markup Language for Exchanging Business Processes? Potential and Limitations. In M. Nüttgens and J. Mendling, editors, *XML4BPM 2004, Proceedings of the 1st GI Workshop XML4BPM, Marburg Germany, March 2004*, pages 43–60, March 2004.
12. F. Leymann and D. Roller. *Production Workflow: Concepts and Techniques*. Prentice-Hall PTR, Upper Saddle River, New Jersey, USA, 1999.
13. Meta Object Facility (MOF) specification. Technical report, OMG, April 2002.
14. W. Pidcock. What are the differences between a vocabulary, a taxonomy, a thesaurus, an ontology, and a meta-model?, 2004. Web Page URL: <http://www.metamodel.com/article.php?story=-20030115211223271>.
15. M. Rosemann and M. zur Muehlen. Evaluation of Workflow Management Systems - a Meta Model Approach. In K. Siau, Y. Wand, and J. Parsons, editors, *2nd CAiSE/IFIP 8.1 International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD '97)*, Barcelona, 1997.
16. A.W. Scheer. *Wirtschaftsinformatik. Studienausgabe. Referenzmodelle für industrielle Geschäftsprozesse*. Springer, Heidelberg, 1997.
17. W. van der Aalst and K. van Hee. *Workflow Management: Models, Methods, and Systems*. Cooperative Information Systems. The MIT Press, 2002.
18. M. Weber and E. Kindler. *The Petri Net Markup Language*, pages 124–144. LNCS 2472. Springer, 2003.
19. Workflow Management Coalition: Terminology & Glossary. Technical Report WfMC-TC-1011, The Workflow Management Coalition (WfMC), February 1999.
20. Workflow management facility specification, v1.2. Tech. report, OMG, April 2000.
21. Workflow Process Definition Interface – XML Process Definition Language. Technical Report WfMC-TC-1025, WfMC, October 2002. Version 1.0.

On the Syntax of Reference Model Configuration – Transforming the C-EPC into Lawful EPC Models¹

Jan Recker¹⁾, Michael Rosemann¹⁾, Wil van der Aalst²⁾, Jan Mendling³⁾

¹⁾ Faculty of Information Technology
Queensland University of Technology
126 Margaret Street, Brisbane QLD 4000, Australia
{j.recker, m.rosemann}@qut.edu.au

²⁾ Eindhoven University of Technology
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
w.m.p.v.d.aalst@tm.tue.nl

³⁾ Department of Information Systems and New Media
Vienna University of Economics and Business Administration
A-1180 Vienna, Austria
jan.mendling@wu-wien.ac.at

Abstract. For Enterprise Systems (ES) to provide support for the business operations of enterprises, they need to be *configured* to fit organizational requirements. ES reference models aim at supporting this task but, up to now, fail in providing adequate conceptual support due to missing configurability of the models themselves. This paper extends the work on *a configurable reference modeling notation*. In previous research we developed an adequate conceptual notation for configurable reference models. This paper considers a *syntactic* perspective of reference model configuration. We discuss the lawful environments of configurable nodes and report about syntactic and semantic implications of model configuration in these environments. We then apply these findings in the *design of a XML-based interchange format for reference model configuration* and discuss its applicability for the *conceptual design of tool support* for the configuration of reference process models, which will facilitate and aid the verification of the syntactical correctness of configured reference process models that may then be mapped to executable process specifications.

Keywords. Process configuration, reference modeling, Enterprise Systems

1 Introduction – Reference Models and Enterprise Systems

Many organizations suffer problems from badly implemented Enterprise Systems (ES) [1]. Both academia and industry state that these problems result from a misalignment gap between business and IT, which, once closed, would lead to signifi-

¹ The research on the C-EPC is financially supported by SAP Research.

cantly improved business performance [2]. The notion of (mis-) alignment primarily embraces the process dimension, *i.e.* the alignment of IT functionality to the actual business processes of an organization. In many cases, it is observed that the system hampers the normal way of handling processes instead of supporting it.

This is even more surprising given the fact that business process orientation as a concept has been a major topic in both academia and practice at least since the 1990's [3, 4]. Alongside this trend, the IS community has experienced the proliferation of an enormous number of process modeling methods, including the Event-Driven Process Chains (EPC) [5], which itself is used within the Enterprise System SAP.

The term Enterprise Systems represents integrated information systems that aim at holistically supporting the operational processes of organizations. Though ES packages are distributed as commercial off-the-shelf (COTS) software, their implementation often results in tremendous configuration efforts. Given the fact that the alignment of “generic” ES solutions to “specific” organization needs denotes a highly complex task, it was found that a model-driven solution would provide a more intuitive approach towards configuring, adapting, and customizing ES software to customer demands. Such a model-driven approach naturally would take on existing ES *reference models*, which have already been developed by ES vendors in order to improve the understandability of their systems.

In the context of Enterprise Systems, such *application reference models*, describing the structure and functionality of software solutions on different levels of conceptual abstraction [6], are of particular interest. Due to their prescriptive nature, *i.e.* application reference models usually depict the complete functionality of the system [7], they are, however, only of limited use to the ES configuration process, mainly due to a lack of conceptual support in the form of a configurable modeling language underlying the reference models.

Addressing this issue, we have been developing a new reference modeling approach which considers the configurable nature of an Enterprise System. The representation language of this approach is called a *Configurable EPC* (C-EPC). While previous research efforts have been focused on the meta model and the notation of C-EPCs [8], this paper discusses syntactical problems of C-EPCs in the light of reference model configuration. The *scope of our paper* is the translation of (configured) C-EPC models into lawful (regular) EPCs. We will show that the application of C-EPC in the process of ES reference model configuration leads to syntactic problems and we will outline an approach how to handle these problems when translating C-EPC models into lawful process models. More specifically, the *aim of our paper* is to outline a XML schema-based approach using the EPC Markup Language (EPML) [9] for the task of syntactical validation of reference process model configuration.

The remainder of our paper is structured as follows: Section 2 presents issues and shortcomings of the EPC notation in the light of reference model configurability and introduces the notion of a configurable reference process modeling technique. Also, it briefly reports on related work in the field of configurative reference modeling. Section 3 discusses semantic and syntactic problems that occur when configuring reference process models. We then present a XML-based specification of C-EPCs on which the design of tool support for syntax validation and automatic model transla-

tion will be based. We briefly summarize in Section 4 and propose conclusions drawn from our work.

2 A Configurable Reference Modeling Language

2.1 On syntax and semantics of EPCs

In order to gain an understanding for the C-EPC notation and to raise awareness of problems we encounter during reference process model translation, we briefly outline the notion of classical EPC models and discuss some issues related to the informal semantics and syntax of EPC.

The EPC language was developed at the University of Saarland, Germany, in collaboration with SAP AG (see [5]). A simple EPC consists of events as passive states, functions as active transformations, and logical connectors that connect events and states through control flow. EPCs have – amongst others – been used for the design of the reference process models in SAP [7].

As discussed quite intensively in academia, see *e.g.* [10-12], the definition of EPC in [5], on which we based our research on the C-EPC language, leads to syntactic and semantic problems. The syntax of EPCs as deployed in our research context can be found in [8]. However, this definition does not cover behavioral aspects of EPCs and thus may contain semantic ambiguities.

For instance, the informal semantics of an OR-join causes confusion as a joining OR-connector may or may not synchronize incoming process flows [10]. While these problems have been approached by academic contributions, see *e.g.* [11, 13, 14], in general the issue of informal semantics of EPCs must at this point be considered an open issue.

Considering such problems in the light of ES configuration, the informal semantics of EPC lead to severe issues: EPC models, which depict those process scenarios that are deemed relevant to a particular organization, need to be translated into executable process specifications, which an Enterprise System can execute at run-time. Or, consider a workflow management system that defines, executes, manages, and controls the business processes based on these models. In whatever case, it is of paramount importance to have syntactically correct, *i.e.* lawful EPC process models as an outcome of the configuration process that may then be translated.

In our research, however, we did not want to further complicate the semantics of (configurable) EPCs but decided to express the semantics of C-EPCs in terms of traditional EPCs. Hence, we seek to validate the behavior of configurable processes through their translation to regular EPCs. Then, any of the formalization approaches mentioned in [11, 13, 14] may be used as a semantic foundation, and we may stop the semantics discussion here. However, later we will need to discuss some semantic implications when translating Configurable EPCs into lawful process models.

2.2 On Configurable Reference Process Models: The C-EPC Notation

Current reference modeling languages lack configuration support. As an example, the SAP reference model [7], which is depicted in the EPC notation, covers in the version 4.6 more than 1,000 business processes and inter-organizational business scenarios. As the main objective of reference models is to streamline the design of particular models, they are coined by the “Design by Reuse” paradigm. To increase their applicability, such models typically not include merely one proposed alternative for conducting business in a certain domain but a range of often mutually exclusive alternatives. Hence it denotes an ‘upperbound’ of process models that may possibly be implemented in a particular enterprise. As an organization might merely favor one of the depicted alternatives, they potentially only refer to a subset of ES functionality to be implemented and accordingly only to a subset of the reference model. Up to today, however, these types of decision cannot be reflected within the ‘upperbound’ reference model due to lacking configuration support of the underlying reference modeling language. Existing reference modeling techniques do not support the highlighting and selection of different (process) configuration alternatives. This lack of expressiveness obviously denotes a major issue for reference model users.

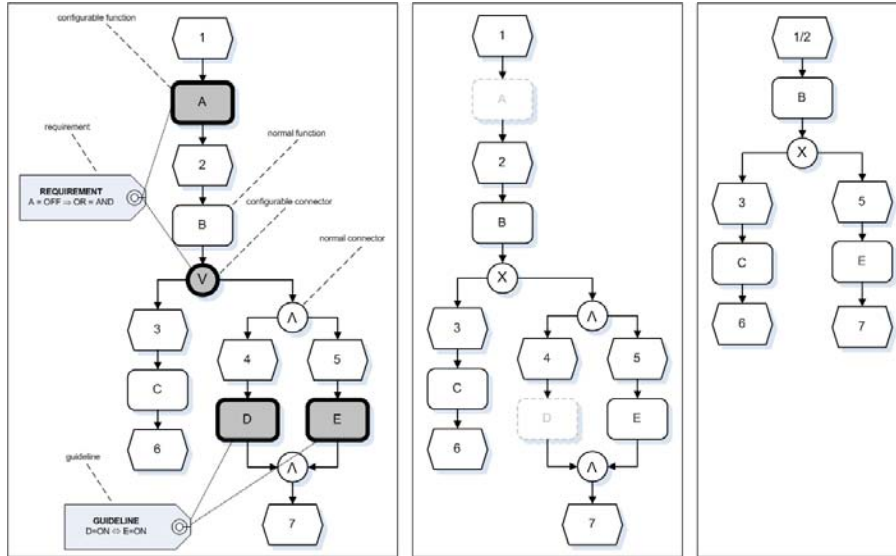


Fig. 1. A simple C-EPC (before configuration, with selected configuration, and resulting EPC)

Addressing these issues, this section introduces *Configurable EPCs* (C-EPCs) (see Fig. 1) as an extension to the popular EPC modeling technique. Focus was spent to the active parts of process models, *i.e.* functionality (functions, tasks, transitions, and the like) and control flow. We have not examined the configurability of events (or states) as more passive parts of processes since they cannot actively be influenced by an organization. It is the reaction to events that can be influenced and this reaction is covered in C-EPCs. The notion of a configurable EPC has been introduced and for-

malized in [8], therefore we only discuss the basic notation here. Fig. 1 shows an example of a C-EPC model, with the left part showing the configuration alternatives, the middle part showing *one* selected configuration alternative, and the right part showing a lawful resulting EPC model based on that configuration.

In a C-EPC functions and connectors can be configured. Notation-wise, these configurable nodes are denoted by thick circles. *Configurable functions* may be included (ON), excluded (OFF), or conditionally skipped (OPT). To be more specific, for configurable functions, a decision has to be made whether to perform this function in every process instance during run time, whether to dispose of this function permanently, *i.e.* it will not be executed in any process instance, or whether to defer this discussion to run time, *i.e.* for each process instance it has to be decided whether to execute the function or not.

Configurable connectors may be restricted. A configurable OR^C -connector may be mapped to a regular OR-, XOR-, or AND-connector. Also, the OR-connector may be disposed of, resulting in a normal process sequence SEQ_i . A configurable AND-connector may only be mapped to a regular AND-connector with a decision being made as to how many of n available process sequences are to be executed in synchronization. A configurable XOR-connector may be mapped to a regular XOR-connector with a decision being made as to how many of n available mutually exclusive process paths are to be provided for execution, or the XOR-connector may be disposed of, resulting in a single process sequence SEQ_i .

In order to depict inter-dependencies between configurable EPC nodes, the notion of *configuration requirements* has been introduced. Inter-related configuration nodes may be limited by these requirements (constraints, expressed denoted by logical expressions). Consider the example given in Fig. 1. If the configurable function A is excluded, the inter-related configurable connector OR^C must be mapped to a regular AND-connector.

Moreover, in order to provide input in terms of recommendations and proposed best practices, *configuration guidelines* may be depicted to guide the configuration process semantically. Consider again the example given in Fig. 1. A recommendation could be that if function D is included, then so should be function E. Summarizing, requirements and guidelines represent hard (*must*) respectively soft (*should*) constraints.

Concluding, we introduced a configurable reference modeling notation which potentially facilitates a model-driven selection and modification of process flows and process activities.

2.3 Related Work

Related work on configurative reference modeling includes the perspectives-based configurative reference process modeling approach by BECKER *et al.* [15]. This approach focuses on adaptation mechanisms and proposes several mechanisms for automatically transforming a reference model into an individual model. While the work of BECKER *et al.* focuses on generic adaptation mechanisms, this research pursues a reference model-driven approach towards ES configuration.

SOFFER *et al.*'s suggestions on ERP modeling [16] can also be regarded as close to our proposed ideas. Following the concept of scenario-based requirements engineering, they evaluate the Object-Process Modeling Methodology in order to determine a most appropriate ERP system representation language. The so-called argumentation facet, related to the ability of a modeling language to express optionality-related information, is just one of many of their criteria. Their work does not comprehensively analyze requirements related to modeling ERP configurability and focuses on technique evaluation rather than on the development of a more appropriate technique.

GULLA and BRASETHVIK [17] introduce three process modeling tiers to manage the complexity of process modeling in comprehensive ERP Systems projects. Their functional tier dimension deals with the functionality of the Enterprise System. However, they do not study how reference models fit into in this tier.

3 On the Syntax of Reference Model Configuration

3.1 Configuration using the C-EPC modeling language

The task of configuring reference models that have been deemed configurable by highlighting variation points embraces both a semantic and a syntactic dimension. While the former is concerned with making business configuration decisions in order to match organizational strategy and requirements, the latter is concerned with maintaining syntactical correctness within the configured models to ensure a lawful translation into executable systems at run time. We will show, that these dimensions are inter-related during configuration as syntactic considerations of implementing the models do have semantic, *i.e.* business consequences and must hence be considered during semantic configuration.

We have described the semantic dimension of configuration in [18]. Simplistically, through the use of the C-EPC notation, process scenarios and process alternatives that are deemed desirable for a particular organization are selected. This is done by switching configurable nodes within a C-EPC model to a desired setting. Configuration requirements and configuration guidelines restrict respectively aid this task.

The outcome of this phase is a C-EPC model where all configurable nodes have been switched to a certain setting. What, however, hasn't been ensured yet, is that these configured C-EPC models apply to the formal syntax of regular EPC. As an example, the middle part of Fig. 1 shows a configured version of the C-EPC model shown in the left part, where the configurable OR-connector has been switched to a regular XOR-connector and where function A and D have been excluded (shaded grey).

As can be seen, the resulting process model would be syntactically inconsistent: Consider function A: Assuming the control flow is reconnected where the excluded function is missing, two events would follow each other. This is syntactically incorrect.

Inadvertently, the step beyond semantic configuration of C-EPC models from a business perspective is the task of re-establishing syntactical correctness and consistency, *i.e.* the translation of configurable process model into lawful regular process specifications (as an example refer to the right part of Fig. 1).

3.2 Translating C-EPCs into EPCs: Syntactical and Semantic Problems

Now, in order to approach the syntactic and inherent semantic problems that arise due to the configuration of C-EPCs, we need to develop a translation approach that maps a configured C-EPC to a lawful regular EPC. This is a delicate task due to the semantic problems of EPCs themselves, as discussed above. There are in principle several options to approach this task:

- Refine the EPC specification to arrive at rigorously and unambiguously defined semantics for EPCs and thus, for C-EPCs.
- Ignore the semantics of EPCs and merely focus on specifying an unambiguous translation of C-EPCs to EPCs which themselves may then be further discussed.

Here, we opted for the latter alternative: We wanted to *extend* the work on reference modeling techniques rather than developing new ones. Due to its popularity for the design of reference models and referring to the extensive academic work on its formalization and definition we deemed it better to take EPCs as both starting and ending point for our design of configurable process models instead of proposing yet another semantic and syntactic definition of EPCs.

Now, looking at the configuration of reference process models, this task can be divided into *global* and *local* decisions, with the former being based on the general model context and which can be made without studying the individual process model. Local decisions on the other hand require an explicit study of the relevant (parts of) process models. Our forthcoming discussion is focusing on the *local* aspects of configuration. We do not deem it necessary to explicitly address global decisions for the following reasons: First, EPCs and thus C-EPCs can be hierarchically structured by decomposing single EPCs into more detailed sub-models. Analogously, each (C-) EPC may be generalized to a simpler EPC on a coarser level of detail. Hence, all contexts of configurable nodes may eventually be drilled up to the smallest possible local environment, as will be discussed below. Second, the notion of C-EPCs provides explicit representation for the depictions of inter-dependencies and inter-relationships between configurable nodes. Hence, global inter-relationships between processes depicted in separate process models may be expressed, thereby not needing an explicit addressing of a global process context. Third, as current practice shows (consider *e.g.* the configuration of the SAP system), the process of reference model configuration starts at a very coarse level of detail with industry sector-spanning process models (in the SAP context: collaborative business scenarios). At this stage, configuration refers to deleting dispensable processes from high-level process models. It can be seen as more of a scoping exercise in a pre-implementation stage. Hence, global configuration decisions merely are decisions as to the inclusion or exclusion of processes, the former of which then need to be *locally* configured.

Concluding, we argue that configured C-EPC models can be transferred into lawful EPC models in accordance to laws based on the *local* syntactic environment of configurable nodes. We must, for the purpose of this paper, limit some of the discussions to examples. A complete discussion of all local environments for configurable nodes and the entire resulting process model variants would exceed the length of this paper and is furthermore deemed unnecessary for making our argument.

Configurable Functions

Firstly, we investigate the local environments of configurable functions. As an EPC consist of events (E), functions (F), and splitting (S) respectively joining (J) connectors, there are nine different local environments for a configurable function A (see Fig. 2).

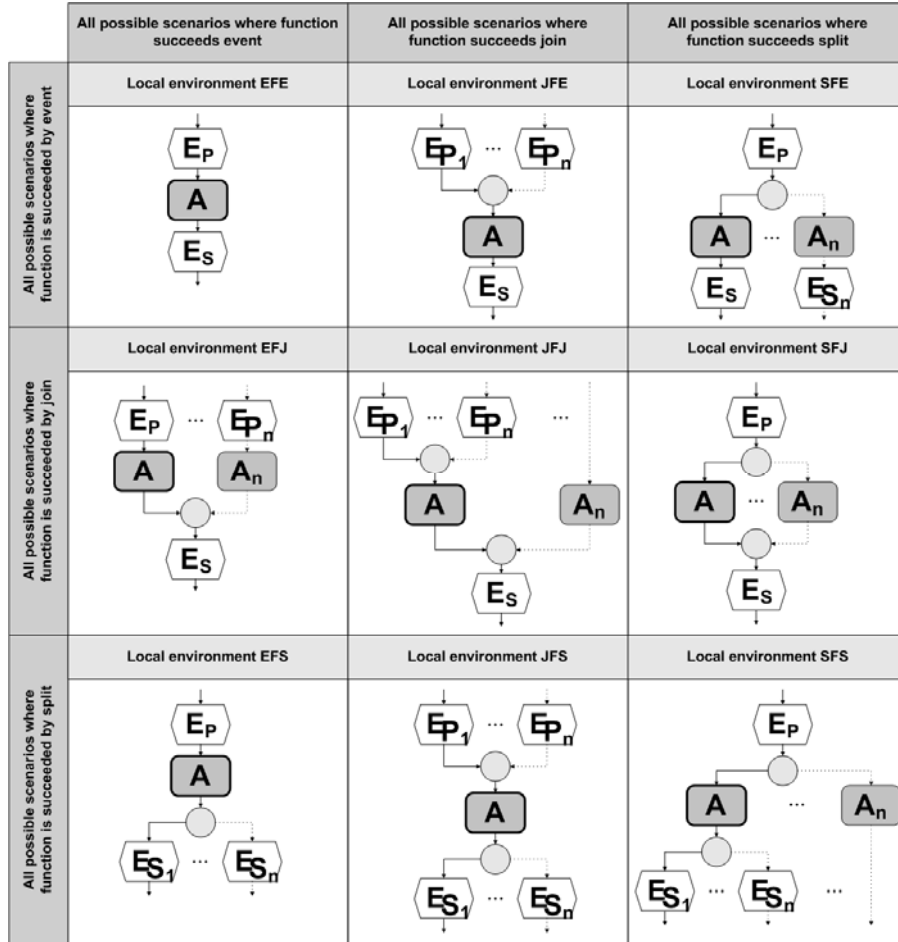


Fig. 2. Local environments for configurable functions

Studying the local environments of configurable functions it becomes obvious that, once a configurable function A has been switched to a desirable setting, the syntactical clean-up of the process model is not purely a technical decision. Due to missing formal semantics of the EPC notation – *e.g.* the EPC modeling language does not explicitly differ between triggering and resulting events that pre-/succeed a function – removals or inclusions of process model elements (such as the disposal of an event that follows a function) may have semantic and thus, business-related consequences.

Bearing that in mind, syntactic validation may lead to various syntactically lawful yet semantically different process models.

Consider the following example. Referring to the local environment ‘Event-Function-Event, EFE’ – the configurable function A is embedded in the context of a preceding event E_P and a succeeding event E_S – configuration and syntactic validation may lead to the process model variants shown in Fig. 3.

Now, as can be seen in Fig. 3, the syntactic handling of switching configurable functions “on” or “off” are simple, according to the definitions in [8]. Optional functions are trickier.

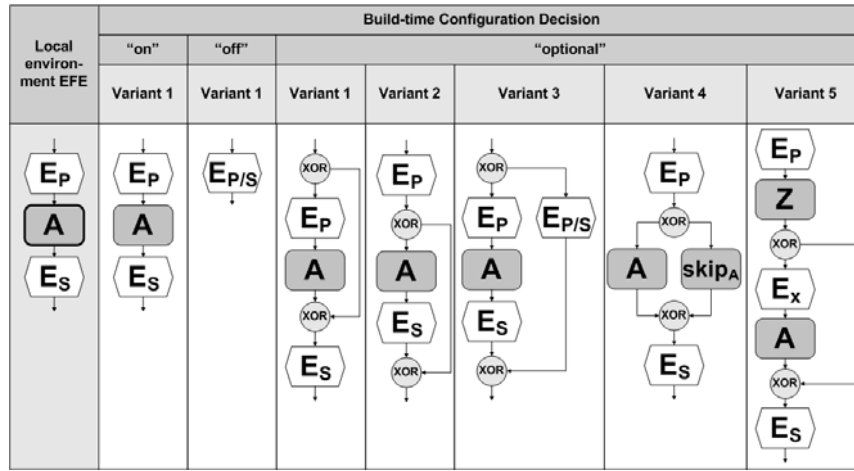


Fig. 3. Lawful alternatives for configuring a function in the EFE environment

Consider the configuration decision of switching the function A to “optional”. The resulting process model must cater for a run time decision to either bypass the function or execute it. Due to the informal EPC semantics, it is not necessarily obvious whether the succeeding event E_S denotes a triggering state for a subsequent business function or simply a resulting state for A. In the former case, the bypass does not need to include E_S (variant 1). In the latter case, E_P needs not to be bypassed (variant 2). Maybe both states surrounding A may be bypassed, thereby passing a new state $E_{P/S}$ (variant 3). Another syntactically valid solution is to introduce a ‘dummy’ function “skipA” which just propagates a process folder from E_P to E_S without any transformation (variant 4). Or, a new decision function Z and an additional event E_x are introduced to augment the configuration decision of switching A to “optional” (variant 5). This case, obviously, requires the inclusion of knowledge external to the model in order to specify the decision function Z.

Configurable connectors

Considering configurable connectors and referring back to the configuration constraints described in Section 2.2, these nodes may appear in any of the local environments shown in Fig. 4.

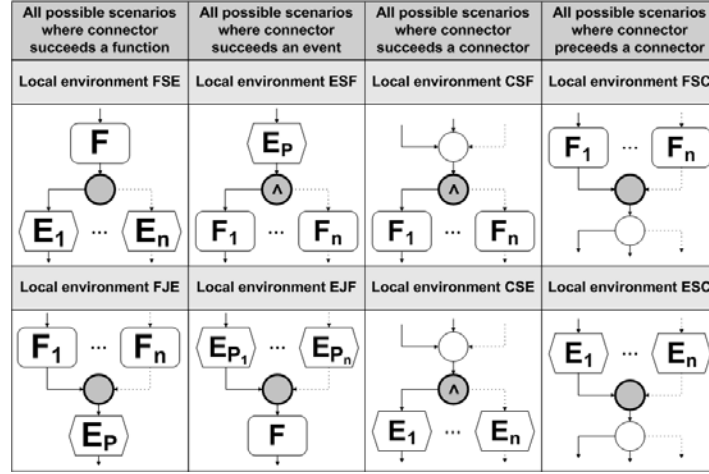


Fig. 4. Local environments for configurable connectors

According to the syntax rules of lawful EPC, some local environments are restricted to the AND connector, since both OR- and XOR-connector need to be linked to a preceding function that allows for the decision which branch to take. With respect to syntactically lawful process variants for these local environments, configurable connectors are easier to handle, as shown in Fig. 5.

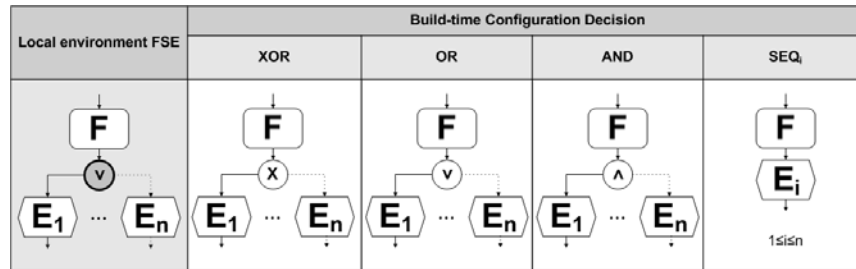


Fig. 5. Lawful alternatives for configuring an OR-connector in the FSE environment

As can be seen, for each configuration decision there exists exactly one syntactic lawful process variant. As can easily be shown, for each of the configurable XOR- and AND-connectors there exists merely one syntactic variant per desired setting as both configurable nodes may only be restricted in their behavior or mapped to a single sequence SEQ_i. Analogously, as configurable connectors are defined to at most restrict their behavior, it is obvious that for each configuration in whatever local environment there can only exist one corresponding syntactically lawful process variant.

Synopsis

The syntactic alternatives for all other local environments of configurable nodes, as depicted in Fig. 2 and Fig. 4 are constructed in a similar way. We examined the lawful environments of configurable nodes and constructed syntactic alternatives for all

combinations of predecessors and successors. As already mentioned, we cannot discuss them in detail here.

As can be shown through our examples, the syntactic clean-up of configured reference process models bears some semantic decisions in itself. Due to the inherent ambiguity of both syntax and semantics of the EPC modeling technique, syntactical validation of C-EPC models may lead to several syntactically lawful yet semantically different process model variants. Since we decided not to modify the EPCs but instead to base our work on the (arguably ambiguous) traditional EPC definition, there results a need for adequate tool support that facilitates and moreover aids the translation process from C-EPCs to EPCs. We will thus, in the next section, address this translation task by presenting a XML-based schema specification of C-EPCs that will be used to aid the syntax validation and translation of C-EPCs to regular lawful process models.

3.3 Towards Tool Support for Reference Model Configuration

Research towards tool support for C-EPCs based on an interchange format was motivated by two facts:

- A configuration of a C-EPC should correspond to a concrete EPC [8]. However, as we discussed in this paper, it is not possible to automate such mapping, hence adequate tool support is needed to facilitate and aid this task.
- EPCs and thus C-EPCs are not executable and thus cannot serve as specifications for process or workflow execution engines – which would, however, be desirable especially in the light of Enterprise Systems. In order to facilitate the interchange of configured reference process models to other process specifications, a standardized interchange format for “cutting-edge” process languages is needed.

Contemplating available options, we deemed a design specification based on a XML schema the best alternative. In particular, we opted for the *EPC Markup Language* (EPML) [9]. This selection was made for the following reasons: First, the EPML is able to perform syntax validations of EPCs [19]. Second, the EPML leverages the interchange of EPCs to other process modeling and execution languages [20], *e.g.* Petri nets. Third, EPML can be generated from the ARIS Markup Language (AML) and is also supported by open source modeling solutions [21], *e.g.* EPC Tools; hence, tool platforms are available for implementing reference model configuration tool support based on C-EPCs.

Now, due to space limitations we cannot give a thorough introduction to the EPML definition, which can be found at <http://wi.wu-wien.ac.at/~mending/EPML/>. Instead, we merely introduce the main extensions to the EPML to cater for the C-EPC specifications (see Table 1).


As can be seen from Table 1, for each configurable node we introduce an EPML representation element. A configurable function is defined as an extension to a regular EPC function in EPML, merely annotating a new attribute element *configuration*, which is optional and may take a value of *on*, *off*, or *opt*. Configurable connectors are likewise specified as extensions to regular connectors, with the option of setting the attribute element *configuration* to a concrete value – in accor-

dance to the definitions outlined in Section 2.2. Specifically, if for a configurable connector the value `seq` is selected, an attribute `goto` specifies the ID of an EPC node of the process model sequence selected. Configuration requirements and guidelines, respectively, are defined as logical expressions involving a number of configurable nodes. In EPML they are thus defined as part of the root `epc` element, with a list containing the IDs of involved elements (`idRefs`). The logical expressions themselves can be modeled via XPath expressions, *e.g.*

```
<configurationRequirement idRefs="2 4">
  <if xpath="function[@id='2']//configuration[@value='off']">
    <then xpath="function[@id='4']//configuration[@value='on']">
  </configurationRequirement>
```

Note that this specification allows for a representation of C-EPCs both *before* configuration (such as the one depicted in the left part of Fig. 1), and *after* configuration (such as the one depicted in the middle part of Fig. 1). Also, as our definitions are mere extensions to the traditional EPC specification in EPML, on the one hand traditional EPC models represented in EPML can also be validated against the extended EPML schema, and on the other hand EPML tools that are not aware of configuration aspects are still able to process C-EPCs as traditional EPCs by simply ignoring the additional configuration element information.

Table 1. EPML representations for C-EPC notation

| C-EPC specification element | EPML representation |
|---|---|
|  | <pre><xs:element name="configurableFunction"> <xs:complexType> <xs:choice minOccurs="0"> <xs:element name="configuration"> <xs:complexType> <xs:attribute name="value" use="optional"> <xs:simpleType> <xs:restriction base="xs:string"> <xs:enumeration value="on"/> <xs:enumeration value="off"/> <xs:enumeration value="opt"/> </xs:restriction> </xs:simpleType> </xs:attribute> </xs:complexType> </xs:element> </xs:choice> </xs:complexType> </xs:element></pre> |



```
<xs:element name="configurableConnector" type="typeCOR">
  <xs:complexType>
    <xs:choice minOccurs="0">
      <xs:element name="configuration">
        <xs:complexType>
          <xs:attribute name="value" use="optional">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:enumeration value="or"/>
                <xs:enumeration value="and"/>
                <xs:enumeration value="xor"/>
                <xs:enumeration value="seq"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="goto" type="xs:integer"/>
        </xs:complexType>
      </xs:element>
    </xs:choice>
  </xs:complexType>
</xs:element>
```



```
<xs:element name="configurableConnector" type="typeCXOR">
  <xs:complexType>
    <xs:choice minOccurs="0">
      <xs:element name="configuration">
        <xs:complexType>
          <xs:attribute name="value" use="optional">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:enumeration value="xor"/>
                <xs:enumeration value="seq"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="goto" type="xs:integer"/>
        </xs:complexType>
      </xs:element>
    </xs:choice>
  </xs:complexType>
</xs:element>
```



```
<xs:element name="configurableConnector" type="typeCAnd">
  <xs:complexType>
    <xs:choice minOccurs="0">
      <xs:element name="configuration">
        <xs:complexType>
          <xs:attribute name="value" default="and"
            use="optional">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:enumeration value="and"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:complexType>
      </xs:element>
    </xs:choice>
  </xs:complexType>
</xs:element>
```



```

<xs:element name="configurationRequirement"
  <xs:complexType>
    <xs:sequence>
      <xs:element name="if">
        <xs:complexType>
          <xs:attribute name="xpath" type="xs:string"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="then" maxOccurs="unbounded">
        <xs:complexType>
          <xs:attribute name="xpath" type="xs:string"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="idRefs">
      <xs:simpleType>
        <xs:list itemType="xs:integer"/>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
</xs:element>

```



```

<xs:element name="configurationGuideline">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="if">
        <xs:complexType>
          <xs:attribute name="xpath" type="xs:string"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="then" maxOccurs="unbounded">
        <xs:complexType>
          <xs:attribute name="xpath" type="xs:string"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="idRefs">
      <xs:simpleType>
        <xs:list itemType="xs:integer"/>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
</xs:element>

```

Now, based on these EPML specifications, reference model configuration tool support may be designed that facilitates the model-driven configuration and translation of C-EPCs. In particular, the EPML specifications will be used to (a) leverage the modeling of C-EPCs via existing modeling tools, such as ARIS or the open source platform EPC Tools, (b) design a XML schema-based tool for checking the validity of configurations, (c) implement an EPML-based program for translating C-EPCs in EPCs, and (d) facilitate the interchange of configured (C-) EPCs to other process specification languages.

4 Summary & Conclusions

This paper reported on syntactical and semantic complications of reference model configuration, using the example of translating C-EPC models to lawful regular EPC models. We argued that configuration occurs before the background of the local environment of the configurable nodes in C-EPC models and showed that both a syntactical and semantic perspective must be considered when mapping configurable nodes

to desired regular EPC nodes. Resulting from these elaborations, we presented our initial conceptual work towards adequate tool support for the configuration of process models. Based on our research, adequate tool support can be designed that embeds our recommendations and thereby guides users when configuring Enterprise Systems based on configurable reference process models.

Our research has a few limitations. First, our conceptual approach needs to be empirically validated to prove its feasibility and applicability. We already conducted an initial laboratory experiment with postgraduate IT students at an Australian university on the perceived usefulness and perceived ease of use of C-EPCs in comparison to EPCs resulting in the finding that C-EPCs are in fact perceived as more useful and easier to use for the task of reference model configuration.² However, we still need to empirically test our work with real business practitioners. This task is currently underway. Second, we focused on the EPC notation and neglected the question of its executability. However, we selected the EPML interchange format as a basis for our conceptual design of tool support for good reason, as it denotes an interchange format for various process modeling languages and may hence facilitate such translation from (C-) EPC models to executable process specifications.

Acknowledgements

We appreciate the continuous fruitful contributions of Alexander Dreiling and Wasim Sadiq to the C-EPC research project and of Markus Nüttgens to the EPML initiative.

References

1. Scott, J.E., Vessey, I.: Managing risks in enterprise systems implementations. *Communications of the ACM* 45 (2002) 74-81
2. Sabherwal, R., Chan, Y.E.: Alignment Between Business and IS Strategies: A Study of Prospectors, Analyzers, and Defenders. *Information Systems Research* 12 (2001) 11-33
3. Davenport, T.H., Short, J.E.: The New Industrial Engineering: Information Technology and Business Process Redesign. *Sloan Management Review* 31 (1990) 11-27
4. Hammer, M., Champy, J.: *Reengineering the Corporation: A Manifesto for Business Revolution*. Harpercollins, New York (1993)
5. Keller, G., Nüttgens, M., Scheer, A.-W.: *Semantische Prozessmodellierung auf der Grundlage "Ereignisgesteuerter Prozessketten (EPK)"*. Arbeitsberichte des Instituts für Wirtschaftsinformatik 89. Institut für Wirtschaftsinformatik der Universität Saarbrücken, Saarbrücken (1992)
6. Rosemann, M.: Using reference models within the enterprise resource planning lifecycle. *Australian Accounting Review* 10 (2000) 19-30
7. Curran, T., Keller, G., Ladd, A.: *SAP R/3 Business Blueprint: Understanding the Business Process Reference Model*. Prentice Hall PTR, Upper Saddle River (1997)
8. Rosemann, M., van der Aalst, W.: *A Configurable Reference Modelling Language*. *Information Systems to appear* (2005)

² The design and outcomes of the laboratory experiment are available from the authors on request.

9. Mendling, J., Nüttgens, M.: EPC Markup Language (EPML) - An XML-Based Interchange Format for Event-Driven Process Chains (EPC). Technical Report JM-2005-03-10. Vienna University of Economics and Business Administration, Vienna (2005)
10. van der Aalst, W., Desel, J., Kindler, E.: On the semantics of EPCs: A vicious circle. In: Nüttgens, M., Rump, F.J. (ed.): Proceedings of the GI-Workshop und Arbeitskreistreffen EPK 2002 - Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten. GI-Arbeitskreis Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, Trier (2002) 71-79
11. van der Aalst, W.: Formalization and Verification of Event-driven Process Chains. Information and Software Technology 41 (1999) 639-650
12. Nüttgens, M., Rump, F.J.: Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK). In: Desel, J., Weske, M. (ed.): Proceedings of the GI-Workshop und Fachgruppentreffen Promise 2002 - Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen. Lecture Notes in Informatics, Vol. P-21. Gesellschaft fuer Informatik, Potsdam (2002) 64-77
13. Langner, P., Schneider, C., Wehler, J.: Petri Net Based Certification of Event-Driven Process Chains. In: Desel, J., Silva, M. (ed.): Proceedings of the 19th International Conference on Application and Theory of Petri Nets. Lecture Notes in Computer Science, Vol. 1420. Springer, Lisbon (1998) 286-305
14. Dehnert, J., Rittgen, P.: Relaxed Soundness of Business Processes. In: Dittrich, K.R., Gerpert, A., Norrie, M.C. (ed.): Proceedings of the 13th International Conference on Advanced Information Systems Engineering. Lecture Notes In Computer Science, Vol. 2068. Springer, Interlaken (2001) 151-170
15. Becker, J., Delfmann, P., Dreiling, A., Knackstedt, R., Kuropka, D.: Configurative Process Modeling - Outlining an Approach to increased Business Process Model Usability. In: Khosrow-Pour, M. (ed.): Proceedings of the 14th Information Resources Management Association International Conference. IRM Press, New Orleans (2004) 615-619
16. Soffer, P., Golany, B., Dori, D.: ERP modeling: a comprehensive approach. Information Systems 28 (2003) 673-690
17. Gulla, J.A., Brasethvik, T.: On the Challenges of Business Modeling in Large-Scale Reengineering Projects. In: Chen, P.P., Embley, D.W., Kouloumdjian, J., Liddle, S.W., Roddick, J.F. (ed.): Proceedings of the 4th International Conference on Requirements Engineering. IEEE, Schaumburg (2000) 17-26
18. Dreiling, A., Rosemann, M., van der Aalst, W., Sadiq, W., Khan, S.: Model-Driven Process Configuration of Enterprise Systems. In: Sinz, E.J., Ferstl, O.K. (ed.): Proceedings of the 7th International Tagung Wirtschaftsinformatik. Gesellschaft für Informatik, Bamberg (2005) 691-710
19. Mendling, J., Nüttgens, M.: EPC Syntax Validation with XML Schema Languages. In: Nüttgens, M., Rump, F.J. (ed.): Proceedings of the 2nd GI Workshop on Event-Driven Process Chains. GI-Arbeitskreis Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, Bamberg (2003) 19-30
20. Mendling, J., Nüttgens, M.: Exchanging EPC Business Process Models with EPML. In: Nüttgens, M., Mendling, J. (ed.): Proceedings of the 1st GI Workshop XML4BPM - XML Interchange Formats for Business Process Management. GI-Arbeitskreis Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, Marburg (2004) 61-79
21. Mendling, J., Nüttgens, M.: Transformation of ARIS Markup Language to EPML. In: Nüttgens, M., Rump, F.J. (ed.): Proceedings of the 3rd GI Workshop on Event-Driven Process Chains. GI-Arbeitskreis Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, Luxembourg (2004) 27-38

Configurable Process Models as a Basis for Reference Modeling

– position paper –

W.M.P. van der Aalst^{1,3}, A. Dreiling^{2,3}, F. Gottschalk¹, M. Rosemann³, and
M.H. Jansen-Vullers¹

¹ Department of Technology Management, Eindhoven University of Technology, P.O.
Box 513, NL-5600 MB, Eindhoven, The Netherlands.

`w.m.p.v.d.aalst@tm.tue.nl`

² European Research Center for Information Systems, University of Münster
Leonardo-Campus 3, 48149 Münster, Germany.

³ Queensland University of Technology, 126 Margaret St, Brisbane, QLD 4000,
Australia.

Abstract. Off-the-shelf packages such as SAP need to be configured to suit the requirements of an organization. Reference models support the configuration of these systems. Existing reference models use rather traditional languages. For example, the SAP reference model uses Event-driven Process Chains (EPCs). Unfortunately, traditional languages like EPCs do not capture the configuration-aspects well. Consider for example the concept of “choice” in the control-flow perspective. Although any process modeling language, including EPCs, offers a choice construct (e.g., the XOR connector in EPCs), a single construct will not be able to capture the time dimension, scope, and impact of a decision. Some decisions are taken at run-time for a single case while other decisions are taken at build-time impacting a whole organization and all current and future cases. This position paper discusses the need for *configurable process models* as a basic building block for reference modeling. The focus is on the control-flow perspective.

1 Introduction

The main objective of reference models is to streamline the design of particular models by providing a generic solution [19]. The application of reference models is motivated by the “Design by Reuse” paradigm. Reference models accelerate the modeling and configuration process by providing a repository of potentially relevant models. These models are ideally “plug and play” but often require some customization/configuration to be adjusted to individual requirements [7]. A configurable process model provides rules defining how a reference model can be adapted. Such a generating adaptation must be distinguished from non-generating adaptations as, e.g., aggregation, specialization or instantiation [5]. Unfortunately, the languages used for reference modeling [4, 8, 18] provide little or no support for configuration. The goal of this position paper is to discuss the need for *configurable process models*.

One of the most comprehensive models is the SAP reference model [8]. Its data model includes more than 4000 entity types and the reference process models cover more than 1000 business processes and inter-organizational business scenarios [19]. Most of the other dominant ERP vendors have similar or alternative approaches towards reference models. Foundational conceptual work for the SAP reference model has been conducted by SAP AG and the IDS Scheer AG in a collaborative research project in the years 1990-1992 [13]. The outcome of this project was the process modeling language Event-Driven Process Chains (EPCs) [13, 14], which has been used for the design of the reference process models in SAP. EPCs also became the core modeling language in the Architecture of Integrated Information Systems (ARIS) [21, 22]. It is now one of the most popular reference modeling languages and has also been used for the design of many SAP-independent reference models (e.g., the ARIS-based reference model for Siebel CRM or industry models for banking, retail, insurance, telecommunication, etc.). *Despite its success, the basic EPC model offers little support for process configuration.* It contains (X)OR connectors but it is unclear whether the corresponding decisions need to be taken at run-time (e.g., based on the stock-level), at build-time (e.g., based on the size of the organization using SAP), or somewhere in-between (e.g., based on the period of the year or resource availability). Therefore, we developed the so-called *Configurable EPCs* (C-EPCs) [19, 9], a generic-monolithic approach for constructing re-usable models [10]. Indeed C-EPCs are extending the configuration opportunities of build-time operators [23, 20, 17]. However, they only provide a partial solution as they are only a representation variation, based on a specific language (EPCs), allowing the user to select or hide elements [5, 6]. In this position paper we would like to trigger a discussion on requirements for configurable process models in a broader perspective.

The remainder of the paper is organized as follows. First, we elaborate on the concept of “choice” which is essential for configurable process models. Second, we approach the problem from a more theoretical viewpoint, i.e., we depict what the essence of configuration is. Finally, we briefly discuss Configurable EPCs as a first step towards such configurable models.

2 Configuration: It is all about making choices

This paper focuses on configurable process models, i.e., we restrict ourselves to the control-flow perspective [12]. There are many languages to model processes ranging from formal (e.g., Petri nets and process algebras such as Pi calculus) to informal (flow charts, activity diagrams, EPCs, etc.). Each of these languages provides some *notion of choice* (e.g., two transitions sharing a single input place in a Petri net or an (X)OR-split connector in an EPC). Typically, it is not possible to describe the nature of such a choice. At best one can either specify a Boolean condition based on some data element (data-based decision) or one can specify events that have to occur for triggering paths (event-based decision) [16]. The usual interpretation is that a choice is made at run-time, based on such

a Boolean condition or based on occurring events. *In the context of reference models, this interpretation is too narrow.*

The *scope* of a decision can vary. For example, if a hospital uses a rule like “If a patient has high blood pressure a day before the planned operation, the operation will be canceled”, then the scope of each choice (operate or not) is limited to a single patient. There may also be choices which affect more cases, e.g., consider the rule “If there is a major disaster in the region, all planned operations will be canceled.” or also an entire process, e.g., “The admittance process requires patients to pre-register.”. There may even be choices that affect all processes in some organizations. The classical process modeling languages, e.g., the languages used in workflow management systems [2, 12], allow only for one level of choices. Reference models have to allow for a broader spectrum of choices. Such choices are called configuration choices and are made at build-time. Configuration choices also affect choices at run-time. For example, at build-time one can choose not to use specific functionality offered by the system. Then no choice needs to be made at run-time anymore. But it may also be possible to use the functionality conditionally (e.g., depending on the workload). In this case the choice must be made at run-time. One can view configuration as *limiting choices by making choices*. Seen from this viewpoint, process modeling languages need to distinguish between run-time choices and configuration choices (i.e., at build-time). Note that the borderline between run-time choices and configuration choices may be a bit fuzzy as the following examples show.

- Based on the volume of the order, the goods are shipped by truck or mail.
- On Saturday, goods are shipped by truck.
- If stock is below 100 items, only preferred customers are serviced.
- The Dutch branches require a deposit, while this is not needed for branches in other countries.
- The organization chooses not to allow for pre-shipments.

Each of these choices is at another level. However, the processes in e.g. the SAP reference model show only one type of choice: the (X)OR-split connector. This triggered us to develop the so-called C-EPCs.

3 Configuration: A theoretical perspective

As described above a reference model provides a generic solution that needs to be configured for a specific situation. A generic-monolithic approach for model reuse should guide the user to a solution fitting to the individual requirements [10]. Therefore the reference model must be able to provide a complete, integrated set of all possible process configurations. This means the reference model is the least common multiple of all process variations, which leads to *inheritance of dynamic behavior* [1, 3]. A reference model can be seen as a subclass of all concrete models. A concrete model itself is a superclass of the reference model. This may create confusion as the term “super” is intuitively connected to the bigger and at first existing reference model (e.g., in [24] traditional inheritance

was altered to depict the reference model as superclass). However, it corresponds to the traditional notion of inheritance in which the subclass adds things to the superclass (e.g., additional methods or attributes). So configuration can be described as the reverse of inheritance. This allows us to use some of the ideas described in [1, 3], in particular we use the idea of *hiding* and *blocking*.

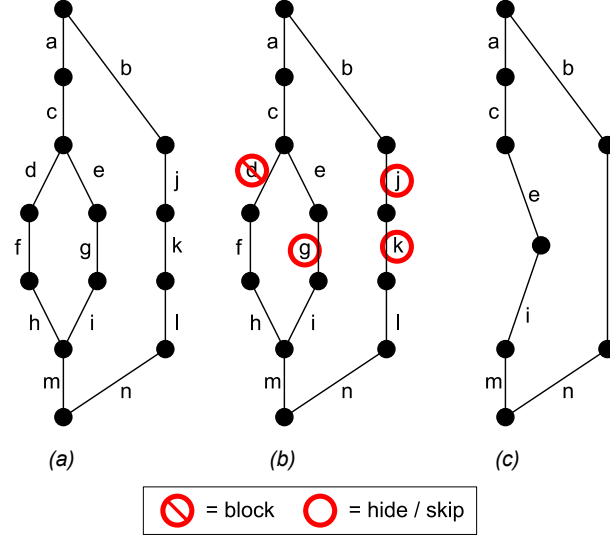


Fig. 1. Three labeled transition systems: (a) the initial model (e.g., the reference model), (b) a particular configuration hiding and blocking specific edges/labels, and (c) the resulting model.

Any process model having formal semantics can be mapped onto a labeled transition system. The nodes in a labeled transition system represent states, the directed edges represent transitions, and each transition has a label denoting some event, action or activity. Traditional choices in the process model, correspond to nodes in the labeled transition system with multiple output arcs. Consider Figure 1(a) showing a labeled transition system. In the initial state (the top node, edges go from top to bottom) there is a choice between a and b . If a is selected, the next step is c and then there is a choice between d and e , etc. If we consider Figure 1(a) to be a reference model, a configuration of this model should select the desired parts. This can be done by blocking and hiding edges or labels. In Figure 1(b) one edge is blocked and three edges are hidden. Hiding and blocking should be interpreted as in [1, 3], i.e., hiding corresponds to *abstraction* and blocking corresponds to *encapsulation*. If an edge is blocked, it cannot be taken anymore. By hiding an edge the path is still possible but the associated label is no longer relevant, i.e., it is renamed to a silent step τ . One can think of the latter as simply skipping the edge. Figure 1(c) shows the resulting model after blocking and hiding the edges indicated in Figure 1(b).

A configurable process model should allow for the specification of which edges/labels can be blocked and hidden/skipped. An interesting question is whether it should be possible to defer this decision to run-time. In the latter case, there would be two more options: *optional blocking* and *optional hiding* (to be decided at run-time).

4 Configuration: An example of a language

To conclude this position paper we introduce *Configurable EPCs* (C-EPCs) as an example for a configurable process modeling language. C-EPCs are an extension of the classical EPCs [13]. A classical EPC consists of functions (i.e., the activities), events and connectors. Functions follow events and events follow functions. Moreover, to model splits and joins in a process connectors may be used. There are three types of connectors: AND, OR and XOR. AND-splits and AND-joins may be used to model parallel routing. XOR-splits and XOR-joins may be used to model the selection of specific routes (e.g., an “if then else” construct). OR-splits and OR-joins may be used to model a mixture of conditional and parallel routing. (However, the semantics of the OR-join is still debated [14].)

In a C-EPC *both functions and connectors may be configurable*. Configurable functions may be included (ON), skipped (OFF) or conditionally skipped (OPT). Configurable connectors may be restricted at build-time, e.g., a configurable connector of type OR may be mapped onto an AND connector. Local configuration choices like skipping a function may be limited by configuration requirements. For example, if one configurable connector c of type OR is mapped onto an XOR connector, then another configurable function f needs to be included. This configuration requirement may be denoted by the logical expression; $c = OR \Rightarrow f = ON$. In addition to these requirements it is possible to add guidelines, supporting the configuration process.

Figure 2 shows a C-EPC describing an invoice verification process. The classical EPC is extended with configurable functions and connectors (indicated using thick lines). For example function *Invoicing Plan Settlement* is configurable, i.e., it may be included (ON), skipped (OFF) or conditionally skipped (OPT). The diagram shows also some configurable connectors. In this position paper we do not further elaborate on C-EPCs. For more information, we refer to [19, 9]. The important thing to note is that it is possible to extend a language like EPCs with configurable elements. Moreover, there are two types of choices: (1) configuration choices made at build-time and (2) “normal” choices made at run-time.

C-EPCs can be seen as a rather naive, but very intuitive, configuration language that allows (optionally) blocking and hiding of edges/labels at build-time for specifying the configuration of the model. Using the theory developed in [1, 3] and basic notions such as simulation, bisimulation, and branching bisimulation [11, 15] on the one hand and practical experiences using C-EPCs on the other hand, we hope to develop more mature configuration languages.

The aim of this position paper is to trigger a discussion on configurable process models. To do this we argued that configuration is strongly related to

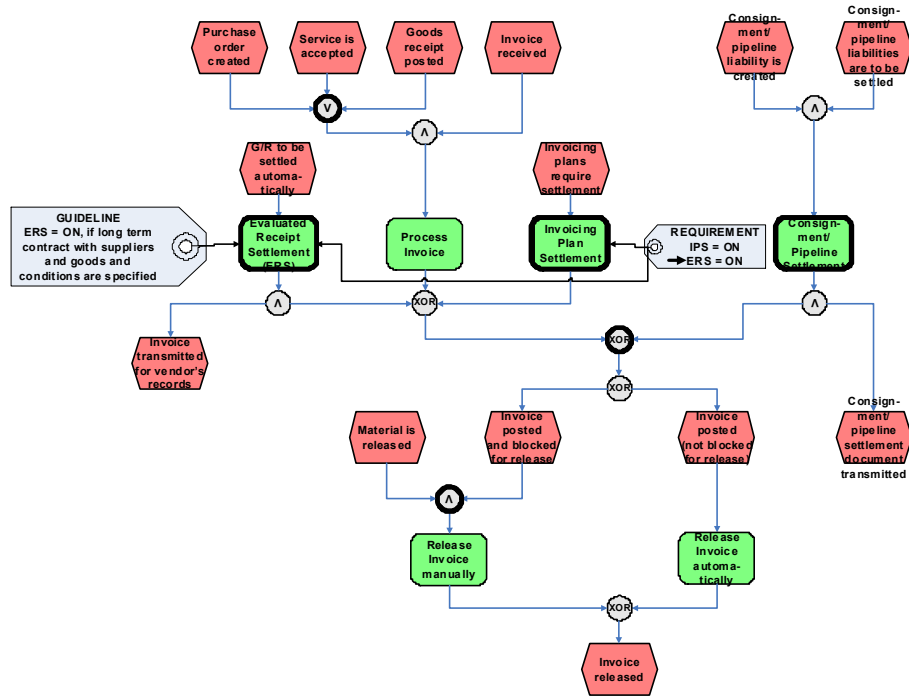


Fig. 2. A Configurable EPC.

the timing and scope of choices. We also showed an example of a language (C-EPCs). However, to allow for a more language-independent discussion we also tried to capture the essence of configuration in terms of (optional) hiding and blocking of edges or labels.

References

1. W.M.P. van der Aalst and T. Basten. Inheritance of Workflows: An Approach to Tackling Problems Related to Change. *Theoretical Computer Science*, 270(1-2):125–203, 2002.
2. W.M.P. van der Aalst and K.M. van Hee. *Workflow Management: Models, Methods, and Systems*. MIT press, Cambridge, MA, 2002.
3. T. Basten and W.M.P. van der Aalst. Inheritance of Behavior. *Journal of Logic and Algebraic Programming*, 47(2):47–145, 2001.
4. J. Becker, M. Kugeler, and M. Rosemann, editors. *Process Management: A Guide for the Design of Business Processes*. Springer-Verlag, Berlin, 2003.
5. J. Becker, P. Delfmann, R. Knackstedt. Konstruktion von Referenzmodellierungssprachen: Ein Ordnungsrahmen zur Spezifikation von Adaptionsmechanismen für Informationsmodelle. In *WIRTSCHAFTSINFORMATIK*, 46(2004)4, pages 251–264.
6. J. Becker, P. Delfmann, A. Dreiling, R. Knackstedt, D. Kuropka. Configurative Process Modeling – Outlining an Approach to increased Business Process Model Usability. In *Proceedings of the 15th Information Resources Management Association International Conference*. New Orleans, 2004.

7. P. Bernus. *Generalised Enterprise Reference Architecture and Methodology, Version 1.6.3*. IFIP/FAC Task Force on Architectures for Enterprise Integration, 1999.
8. T. Curran and G. Keller. *SAP R/3 Business Blueprint: Understanding the Business Process Reference Model*. Upper Saddle River, 1997.
9. A. Dreiling, M. Rosemann, W.M.P. van der Aalst, W. Sadiq, and S. Khan. *Model-driven process configuration of enterprise systems*. In O.K. Ferstl, E.J. Sinz, S. Eckert, and T. Isselhorst, editors, *Wirtschaftsinformatik 2005. eEconomy, eGovernment, eSociety*, pages 687–706, Physica-Verlag, Heidelberg, 2005.
10. P. Fettke and P. Loos. *Methoden zur Wiederverwendung von Referenzmodellen – Übersicht und Taxonomie*. In J. Becker, R. Knackstedt, editors, *Referenzmodellierung 2002: Methoden – Modelle – Erfahrungen*, Arbeitsberichte des Instituts für Wirtschaftsinformatik Nr. 90 (in German), pages 9–33. University of Münster, Münster, 2002.
11. R.J. van Glabbeek and W.P. Weijland. *Branching Time and Abstraction in Bisimulation Semantics*. In *Journal of the ACM*, 43(3):555–600, 1996.
12. S. Jablonski and C. Bussler. *Workflow Management: Modeling Concepts, Architecture, and Implementation*. International Thomson Computer Press, London, UK, 1996.
13. G. Keller, M. Nüttgens, and A.W. Scheer. *Semantische Prozessmodellierung auf der Grundlage Ereignisgesteuerter Prozessketten (EPK)*. Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 89 (in German), University of Saarland, Saarbrücken, 1992.
14. E. Kindler. *On the Semantics of EPCs: A Framework for Resolving the Vicious Circle*. In J. Desel, B. Pernici, and M. Weske, editors, *International Conference on Business Process Management (BPM 2004)*, volume 3080 of *Lecture Notes in Computer Science*, pages 82–97. Springer-Verlag, Berlin, 2004.
15. R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1980.
16. M. Owen and J. Raj. *BPMN and Business Process Management – Introduction to the New Business Process Modeling Standard*, Popkin Software, 2003.
17. M. Rosemann. *Komplexitätsmanagement in Prozessmodellen: methodenspezifische Gestaltungsempfehlungen für die Informationsmodellierung (in German)*. Gabler, Wiesbaden, 1996.
18. M. Rosemann. *Application Reference Models and Building Blocks for Management and Control (ERP Systems)*. In P. Bernus, L. Nemes, and G. Schmidt, editors, *Handbook on Enterprise Architecture*, pages 596–616. Springer-Verlag, Berlin, 2003.
19. M. Rosemann and W.M.P. van der Aalst. *A Configurable Reference Modelling Language*. In *Information Systems* (to appear, also available from BPMCenter.org), 2005.
20. M. Rosemann and R. Schütte. *Grundsätze ordnungsmäßiger Referenzmodellierung*. In J. Becker, M. Rosemann, R. Schütte, editors, *Entwicklungsstand und Perspektiven der Referenzmodellierung*, Arbeitsberichte des Instituts für Wirtschaftsinformatik Nr. 52 (in German), pages 16–33. University of Münster, Münster, 1997.
21. A.W. Scheer. *Business Process Engineering, Reference Models for Industrial Enterprises*. Springer-Verlag, Berlin, 1994.
22. A.W. Scheer. *ARIS: Business Process Modelling*. Springer-Verlag, Berlin, 2000.
23. R. Schütte. *Grundsätze ordnungsmäßiger Referenzmodellierung – Konstruktion konfigurations- und anpassungsorientierter Modelle (in German)*. Gabler, Wiesbaden, 1998.
24. A. Schwegmann. *Objektorientierte Referenzmodellierung: theoretische Grundlagen und praktische Anwendung (in German)*. Gabler, Wiesbaden, 1999.