DE GRUYTER
OPEN

DOI: 10.1515/ijek-2015-0007

AN APPROACH TO E-WORKFLOW SYSTEMS WITH THE USE OF PATTERNS

John Ndeta[1], Stamatia A. Katriou[2], and Kerstin V. Siakas[2]

[1]London Metropolitan University, Department of Computing, Communication Technology & Mathematics, 166-220 Holloway Road, N7 8DB London, UK, jon014@londonmet.ac.uk

[2]Alexander Technological Institute of Thessaloniki, Department of Informatics, P.O. Box 141, GR-57400 Sindos, Greece {stankat, siaka} @it.teithe.gr

ABSTRACT

In today's highly competitive and rapidly changing environment, e-businesses constantly have to modify their business processes, i.e. the flow of documents and tasks in a business also known as workflow. More flexible Workflow Management Systems are required to support these constantly changing processes. In this research a platform independent architecture for the design of e-workflow systems is illustrated. The architecture includes an information pool, namely a Workflow Pattern Repository, which contains patterns, which are repeatable solutions to reoccurring problems, in order to make the system more apt to change and assist the workflow designer/user in defining workflows faster and more accurately. The patterns in the repository are in the form of UML activity diagram templates. A straightforward input format for storing patterns in the repository is provided along with an example of its practical application.

KEYWORDS

Workflow, e-workflow, e-business, Workflow Design Patterns, Pattern Repository, Workflow Architecture, Exceptions.

INTRODUCTION

In today's globalised consumer society, businesses continuously have to adapt to changes in their environment in order to remain competitive or even survive. The phenomenon is even greater in e-business where technology and trends change even more rapidly. E-companies have to manage frequent organisational change and alter their business processes. This means they must constantly modify their workflow which requires flexible workflow management systems.

A workflow management system "completely defines, manages and executes 'workflows' through the execution of software whose order of execution is driven by a computer representation of the workflow logic"[1]. A workflow or workflow model is a definite description of a business process represented in such a way that it can be directly executed by a workflow management system [2].

Workflow systems embody explicit process and product models, i.e., a completely specified workflow design is required that can be modified to reflect the changes in organizations whenever they occur. A major limitation of traditional workflow systems is that they can, typically, only support simple, static and predictable processes, but not the dynamically changing and complex processes that are present in many contemporary e-business organisations [3]. This drawback of traditional workflow systems is an evident fact in modern businesses where workflows are often executed simultaneously, requiring interaction between them and where problems arise during the execution of a workflow that have to be handled properly.

Consequently e-workflow modelling must provide process support like traditional workflows do, but in such a way that the system is intelligent enough to deal with the new Internet business environment that is characterised by rapid, dynamic and discontinuous change. Previous research for tackling the limitations of traditional workflow systems has been presented in brief in a previous paper [3, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]. Our proposed approach to a flexible e-workflow modelling system is based on the extensive use of design patterns.

Workflow design patterns can be seen as generalised solutions to recurring problems within the context of e-business. Tried and tested solutions can be reused to solve recurring problems within e-business environment that is characterised by many uncertainties and variations [5]. Patterns can describe control flow, data, resources or even exceptions. Exceptions are unexpected problems that may occur i.e. unexpected breakdown during the execution of a workflow process, a deadline expiry and any type of change in a business's environment. Workflow systems should support all four types of patterns and have the ability to allow them to interact. This way business processes would be more accurately defined and at the same time they would be easier to change.

This paper presents the proposed architecture of an e-workflow modelling system, accompanied by a description of the types of patterns it supports. There is also an example of a defined pattern in the pattern repository.

1 WORKFLOW AND ENTERPRISE RESOURCE PLANNING SYSTEMS (ERP)

In order to survive in today's global and knowledge economy contemporary e-business organizations and infrastructures require the support of critical business processes. Business processes are generally known to be the fundamental building blocks of an organization's success, and information technologies that focus and support process management and improvement have been successful in helping organizations to meet their corporate missions and to improve their competitive positions. For the last two decades, special interest has been directed towards two distinct technological solutions that improve business processes: Workflow Management (WfM) and Enterprise Resource Planning (ERP) systems. Both classes of systems focus on business processes, but the approaches taken by them are different [21]. A WfMS is implemented based on a process specification and execution paradigm. During the design and implementation of a WfMS, a workflow model is first created to specify organizational business processes, and then workflow instances are created to carry out the actual steps described in the workflow model. During workflow execution, the workflow instances can invoke legacy systems, databases, applications, and can interact with users [22]. ERP systems are often implemented around the idea of off-the-shelf applications [21]. To achieve better "fit" between the off-the-shelf applications and the needs of the organization, ERP systems must be configured by setting various application parameters. The more parameters an ERP application has, the more flexibility in configuring the business process [21]. However, the workflow model in conventional ERP systems is not explicitly specified because it is embedded in the applications and the parameter tables. One way to better understand these differences is to distinguish between flow logic and function logic. Function logic deals with a specific task, such as updating a customer record or calculating order discounts, while flow logic deals with combining many functions in some sequence to solve more complex problems such as processing an order. In ERP systems, flow logic and function logic are both embedded in applications and parameter tables. In contrast, a WfMS separates the two explicitly. Flow logic is captured in a workflow model, usually graphically represented, and function logic is captured in the applications, data, and people the model invokes. Thus, a WfMS enable developers to separate the flows among a system's components (applications, data, people) from the workflow model [22]. Workflow systems are process-centric, focusing on the management of flow logic. On the other hand, ERP systems are data-centric, focusing on managing function logic via a common homogeneous data infrastructure across the organization to support multiple applications.

The main goal of ERP applications is to provide an integrated solution to all business functions (financial, sales, human resource, *etc.*). The underpinning of shared data structures across many applications eliminates the need to pass data step-by-step among applications by accessing data from a common structure. ERP

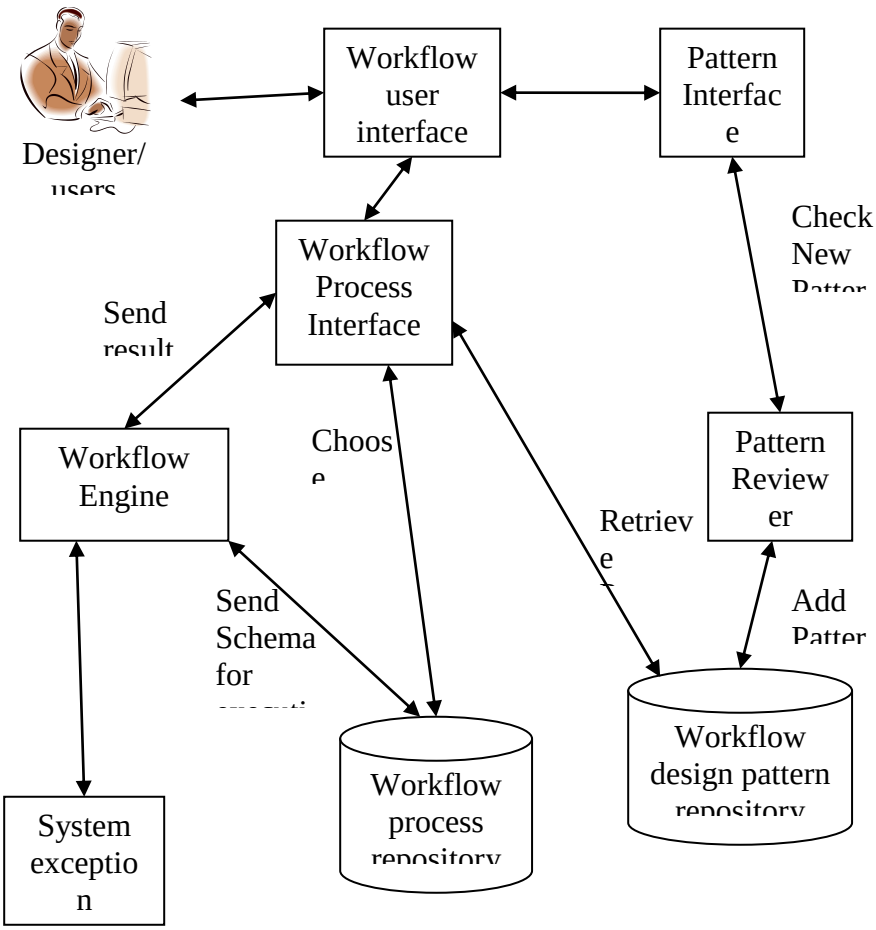
modules operate directly with common interoperable databases to ensure consistent information for all business functions. This makes the manipulation of data easy. During implementation, only data integration from interoperable databases needs to be considered. ERP systems are data-centric, and thus they are well suited for modeling transactional processes for which only data integration is needed. WfMSs are most suitable for modeling workflows involving humans and software systems, especially if the systems are autonomous and heterogeneous.

2. THE PROPOSED E-WORKFLOW SYSTEM ARCHITECTURE

Traditional approaches to workflow modelling mainly deal with the use of rules in specific contexts serving as a mechanism for workflow enactment and evolution [4]. In an e-business environment, focusing on specific context is extremely hard and impractical. Workflow modelling must be as flexible as possible and enable the implementation of a broad range of business processes. In this research we propose a platform independent architecture that uses patterns for workflow modelling. Patterns offer an abstract representation of knowledge and experience from good workflow designers. They can be reused to solve new problems. A workflow system that uses patterns requires a repository to store them and must be equipped with efficient retrieval and adaptation techniques to manage them. This becomes a base for long-term process improvements resulting in increased competitive advantage for e-business enterprises [3].

Figure 1, presents the proposed architecture for an e-workflow system and explains how it functions.

Fig. 1 **Workflow System Architecture (adapted from Ndeta et al. [3])**



Note that workflow process and schema are used interchangeably in this research. From within the workflow user interface the knowledge designer or user can interactively search the workflow design pattern repository or workflow process/schema repository. When a workflow process needs to be defined, the first step is creating a workflow schema. The designer can either create his own description of the workflow process from scratch or he can be assisted by process related knowledge in the form of workflow patterns from the knowledge workflow design pattern repository which provide predefined solutions for common problems. In order to do this, by using the Pattern Interface, he can browse through the categories of pattern templates that reside in the Workflow Design Pattern Repository. Once he has found the right pattern/patterns he can instantiate and customise them so that they will fit his needs. This modified pattern can also be sent to the Workflow Process Repository. Here the modified pattern, called a

workflow schema, is store so that it may be executed by the Workflow Engine when the user wishes to do so.

The user can access the workflow schemas he wishes through the Workflow Process Interface, which allows him to browse the workflow schemas that are in the Workflow Process Repository. With the help of this interface, the user may delete workflow schemas or send them to the Pattern Interface in order for them to be modified. If the user wants to execute a schema then, the selected workflow schema is then sent to the workflow engine for execution. The results are sent to the User Interface.

E-workflow execution may involve the modification of system and workflow relevant data, which can only be accessed by the e-workflow engine and not the pattern component, which is not involved with workflow instances and execution [3].
During the execution, problems may arise that have or haven't been considered in advance. These problems force the workflow to deviate from its defined state and are known as exceptions. Exceptions in the business process e.g. deadline expiry can be described in patterns and incorporated into the workflow schema to avoid problems. However, exceptions such as system breakdowns that have to do with the Workflow System and not the business process must also be handled. Designing alternative actions within the workflow schema for these exceptions would be a waste of designers' time. Therefore, there is a System Exception Handler that can catch and handle all possible technical problems.

Designers can also add new patterns to the workflow design pattern repository. After defining the pattern through the Pattern Interface, they may send the pattern to the repository. Before the pattern is stored, it must be reviewed in order to detect possible flaws in its format. This is the job of the Pattern Reviewer. The reviewer examines the input format of the pattern and its template. If the pattern is approved by the reviewer then it is stored in the workflow design pattern repository. On the other hand, if the reviewer finds possible mistakes or incompatibility problems, appropriate error or warning messages are sent to the designer. By using the Pattern Interface, the designer may also delete or modify any existing patterns. Before any modifications are accepted, they must be sent through the Pattern Reviewer.

2.1 Workflow Design Patterns

Workflow Patterns are used to describe business processes in such a way that they can then be executed by a workflow engine. In other words, they are used to create workflow schemas. Most problems in business processes modelling reoccur. This means that deviations of one pattern can be used to represent many business processes, since they often have similarities. By using patterns, the design of workflows becomes a faster and easier job. When a workflow designer is trying to

describe a common business process, he doesn't have to reinvent the wheel. Instead, he can use workflow patterns. Patterns encapsulate typical rules or a set of rules that capture the knowledge about the occurrence of an exceptional situation and the actions that can be performed to deal with it. They consist of predefined parts, parameterised parts, and optional parts and are viewed as a description of a problem, of a solution, and of the context in which this solution works [3].

Workflow patterns can be categorised, depending on the perspective used to view the workflow specification. Workflow control patterns describe activities in their execution order, i.e. sequential, parallel. An activity or task corresponds to a single unit of work. Workflow data patterns capture the various ways in which data is represented and utilised in workflows. Business documents and other data can flow, in a separate data channel, between business tasks allowing data to be transferred from one workflow to another and can qualify as pre or post conditions for other tasks. Workflow resource patterns depict the various ways that resources are used in workflows, such as the role of people or devices which are responsible for executing tasks [6, 7, 2].

Another important category of patterns are workflow exception patterns. During business processes many unexpected events may happen, i.e. system failure, running out of resources. These events are referred to as exceptions and can be described by certain patterns. The Workflow Management System should have handlers to deal with these exceptions. However, it must be noted that handlers can only be provided for expected exceptions. The workflow designer must incorporate all possible exceptions that may arise in the business process into his design. This should be done in a separate workflow schema so as not to overburden the original business process [8, 9]. Exceptions that are linked with system failure need not be described by the workflow designer, but they should be handled during execution time by a system exception handler.

Research on the formalisation of the above patterns has been done. However, in a real system, process, data and resource perspectives interplay. Therefore considering these patterns in isolation is not sufficient. Attempts to formalise the patterns by combining several perspectives have yet to be made [10].

2.2 The Workflow Design Pattern Repository

As mentioned earlier, the workflow design pattern repository serves as a pool of information to the designer or user, who can interactively search the repository for a suitable workflow pattern. This saves time and promises more accurate solutions to reoccurring problems.

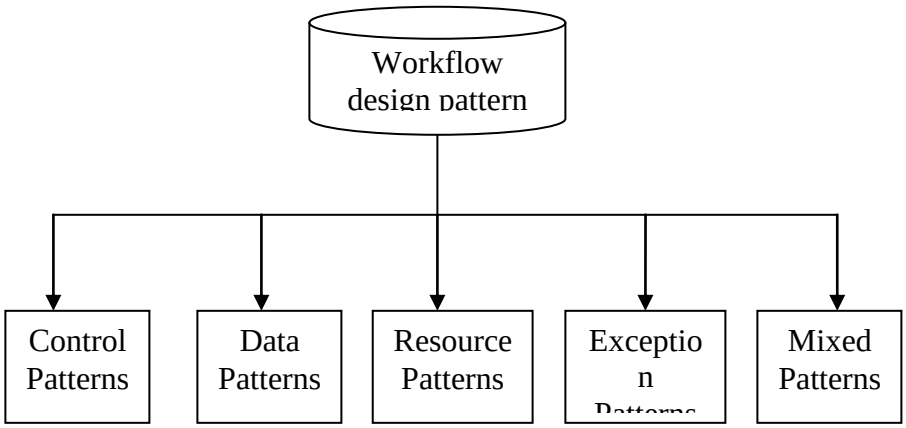
The proposed workflow design repository contains control, data, resource and exception patterns, in the form of abstract templates, that have already been

formalised in others' research. However, business processes are more complex than these simple formalisations that are all isolated from each other. There is a need for patterns that cover more than one perspective. Therefore the repository will also contain complete design patterns that describe whole business processes in an abstract template form, referred to as mixed patterns. The contents of the pattern repository are depicted in Figure 2.

When a workflow designer or user is faced with a problem, he can look up a solution for the problem at hand from the workflow design pattern repository. If the problem, or part of the problem, corresponds to a mixed pattern that is already in the repository then the designer can use the template and easily adapt the pattern to his needs. There may also be multiple workflow design patterns referring to one task, from which the designer will have to choose the one he considers most appropriate. If however, the problem the designer is solving happens to be something new, he can retrieve the simple formalised patterns (control, data, resource, exceptions patterns) that are in the repository and use them as a base for solving his problem. Either way less time and effort is spent in solving the problem and ensuring the soundness of the solution.

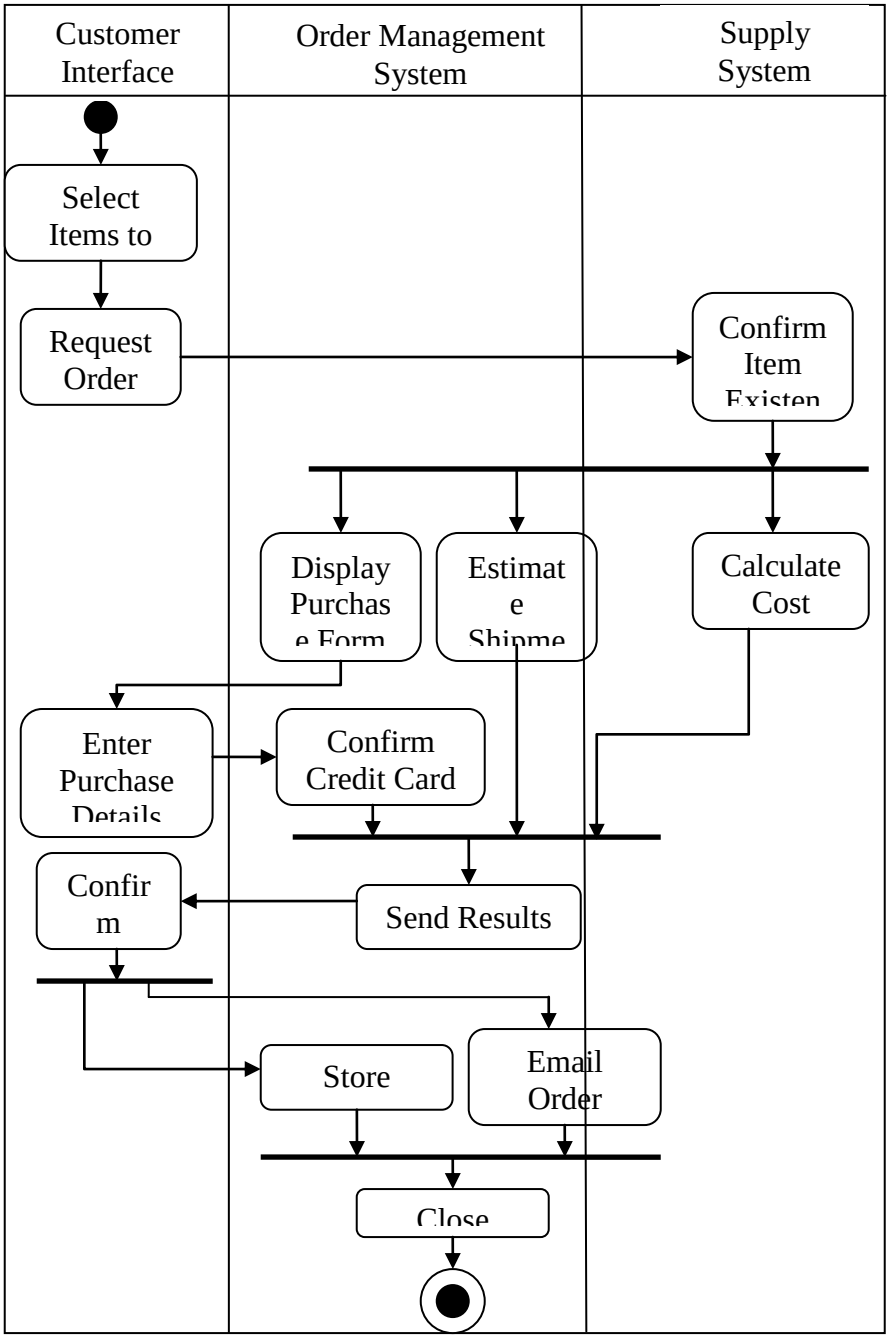
In order to help the designer choose the correct pattern, each time a pattern is used its history will be updated with information about the process it was used for. This way the pattern will become related to certain process structures and the designer will know for which tasks it is suitable.

Fig. 2 **Workflow Design Pattern Repository contents**



The patterns in the repository shall be implemented in the Unified Modelling Language (UML), since it allows the modelling of the various perspectives i.e. process, resource and data perspective.
Each workflow design pattern specifies a set of tasks, together with the constraints and object flows between them. Thus, a workflow design pattern represents one possible means of achieving a given type of task by breaking it down into a particular structure of sub-tasks. Each workflow design pattern specifies a single level of structural decomposition. However, this decomposition refers to a further set of tasks, each of which may have corresponding workflow design patterns existing in the repository. These workflow design patterns may in turn be selected to specialise/instantiate the sub-tasks, and so a multi-level hierarchical process structure may be generated by the composition of many design patterns. Note that for any given task, there may be multiple possible workflow design patterns, expressing different ways of breaking the task down for different situations [3].

Fig. 3 **E-shop Order Processing Workflow**



The UML activity diagram, shown in Figure 3, depicts an order processing workflow of an e-shop. This workflow diagram can not stand alone. It is associated with the workflow of a bank so that the credit card can be verified. The

confirmation of the items' existence belongs to the Supply System that has its own way of working. Even the estimation of the arrival time of products could be represented by its own diagram. These three cases comprise the sub-workflows of the initial order processing workflow. Each needs to be defined with its own design pattern. In turn each of those patterns could depend on other workflows which would have to be separately defined as sub-workflows of the sub-workflow and so on.

The order processing workflow and its sub-workflows can be stored in the pattern repository as mixed pattern templates. These templates are related, so they must be linked to each other in some way, therefore requiring the pattern repository to maintain a history of process structure [3]. Templates should not only be linked in a vertical hierarchy, but also horizontally, meaning that one workflow's sub-workflow should be linked to the corresponding sub-workflows of similar workflows. This helps the designer who is searching for a way to implement an order process. Instead of having to search for patterns that correspond to the sub-workflows, he can immediately see which templates are related and choose the ones he wishes to use.

It is also important that each new pattern in the repository holds a sufficient description of when it is supposed to be used. This serves as a guideline to the designer who will have found many order processing pattern templates in the repository and will need to pick the most suitable one for his needs. As mentioned earlier, this description will be updated as the pattern is reused.

2.3 Format of Patterns in the Repository

In order to store pattern templates in the repository, the Pattern Interface should present the designer with a form of fields to fill in that will help define the pattern and assist in its correct indexing within the repository, thus avoiding the existence of unstructured and incoherent patterns. This pattern input format should be such, that when designers who want to reuse the pattern retrieve it, they will be provided with information about how and when the use of this pattern is appropriate. Based on the pattern format of Ndeta, et al. [3], Muylar and van der Aalst [10] and on already formalised patterns [2, 6, 7, 8] we decided that the following format would be suitable:

– *Name:* a concise name that uniquely identifies the pattern in the workflow design pattern repository. The name must relate to what the pattern does.
– *Author:* names the author and co-authors of the pattern.
– *Type:* refers to the type of pattern, i.e. control, data, resource, exception or mixed pattern.

– *Related Patterns:* names patterns that solve exactly the same problem, thus linking them horizontally into a structure that can be easily indexed.
– *Related sub-patterns:* contains the sub-workflow patterns of the current workflow and the sub-workflow patterns of the previous field.
– *Similar Patterns:* refers to patterns that solve similar problems and could possibly be of interest to a designer who trying to solve the current problem.
– *Classification:* allow users to browse by category, according to the categories and sub-categories of the workflow design pattern repository.
– *Keywords:* a set of user-selected terms that can be used to refer (i.e. select, search, etc) to the available patterns in the repository. This field allows one to give a more precise description of the topics of the pattern and helps distinguish the different patterns of a given category in the classification.
– *User Problem:* a description of the problem as the end user will see it i.e. what is the problem presented to the end user?
– *Intent:* briefly describes the main goal of a pattern, i.e. towards which problem it offers a solution.
– *Problem Description:* gives a detailed presentation of the problem.
– *Solution:* describes possible solutions to the problem
– *Guideline:* provides suggestions to the designer about possible usage and instantiation of patterns
– *Template:* contains the pattern template in the form of a UML activity diagram. The diagram is described in terms of events, conditions, and actions. Unlike events and conditions, which are the main parts of the pattern, the action part provides only suggestions. This reflects the fact that exception patterns focus on how to capture exceptions, rather than on how to fix reactions, which are application dependent. The template contains parametric fields to be filled in with specific values provided by the designer [4].
– *Sample usage:* a set of workflow-specific instantiations of patterns related to an application domain. The instantiations of the pattern template will show examples of how the template can be used. They show how patterns can be customised in different context and applications by illustrating how parameters of patterns can be supplied by the designer to produce a concrete workflow model [4]. The sample templates may be accompanied by an explanation of how they work.

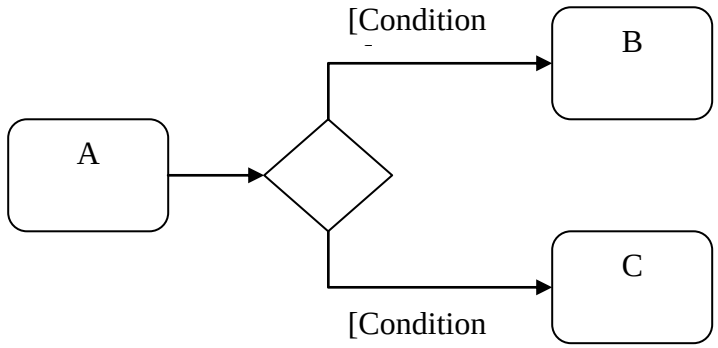
The mandatory fields to define a pattern are the Name, Type, Intent and Template fields. For simple patterns the field Problem Description is unnecessary since the problem is sufficiently described by the Intent field and the Solution field need not be filled in since the template offers a clear description of the solution. However, for mixed patterns the Problem Description and Solution fields should be compulsory, since the template might be too complex to understand without some written guidance. When a field is not filled in it won't show up on the pattern detail page.

Below is an example of the input format with its fields filled in. The pattern chosen is a simple, formalised control pattern.

– *Pattern Name:* Exclusive Choice
– *Author:* Wil van der Aalst, Athur ter Hofstede, Bartek Kiepuszewski, Alister Barros
– *Type:* control
– *Similar Patterns:* Parallel split, Multiple Choice
– *Classification:* simple control pattern
– *Keywords:* XOR, XOR-split, conditional routing, EOR, switch, decision, choose, either
– *Problem:* Choose one of many paths
– *Intent:* Allows the workflow process to split into one of several branches, depending on a decision or workflow data,
– *Guideline:* as many branches as needed may be added to the pattern

***Fig. 4* XOR template**

– *Template:*



– *Sample usage:*

Fig. 5. XOR with three branches for order confirmation

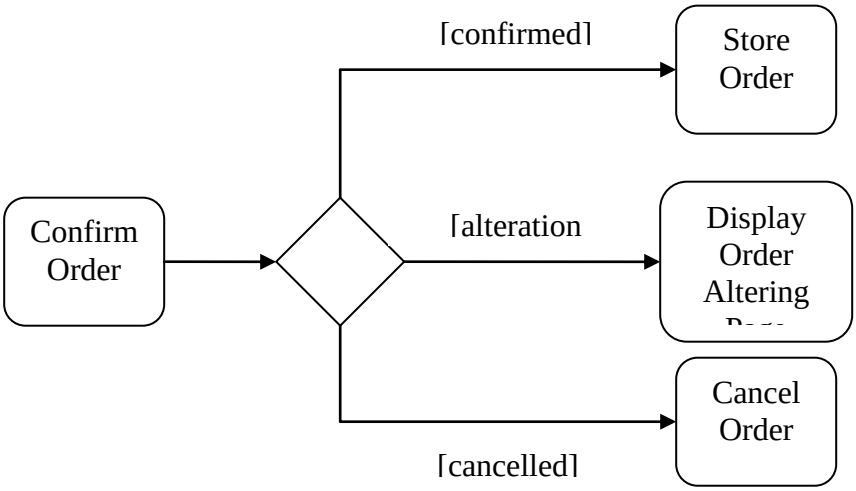
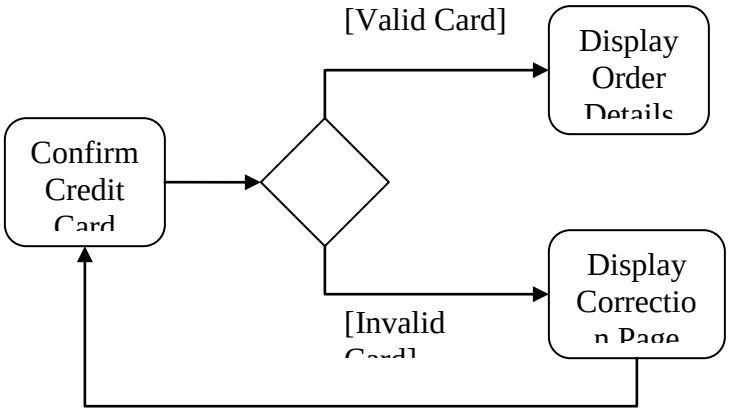


Fig. 6 **XOR for cofirming credit card validity**



3. CONCLUSIVE REMARKS

The continuously changing environment of the internet creates a need for flexible business processes that can be modified frequently, thus augmenting the requirements of Workflow Systems. By creating a system with a reliable exception

handler, a good pattern repository and a proper interface for accessing and managing patterns, workflow designers and users will find it much easier to define and execute appropriate workflow schemas.

The pattern repository must contain simple patterns that are used in all workflow definitions as well as complex patterns. The complex/mixed patterns must be well described so as not to confuse the designer/user. All these patterns are sent to the repository through the input format for pattern creation, which enables patterns' content in the workflow design pattern repository to be structured and predictable. Every time the pattern is used, it must be updated with information about the process it was used for, thus helping designers when they wish to reuse it.
Future research in this field needs to consider formalising basic mixed patterns, designing patterns for authorisation management in Internet workflow and the design of a system exception handler.

REFERENCES

1. Hollingsworth, D.: (Report No. WFMC-TC00-1003) (1995)
2. Russell, N., ter Hofstede, A.H.M., Edmond, D., van der Aalst, W.M.P.: Workflow Resource Patterns. BETA Working Paper Series, WP 127, Eindhoven University of Technology, Eindhoven (2004)
3. Ndeta, J., Marir, F., Choudhury, I.: Knowledge enhanced e-workflow modelling – a pattern- based approach for the development of Internet workflow systems. In Peter Feher (ed), Proceedings of 7th European Conference of Knowledge Management (ECKM06), Budapest, Public Academic Conferences Ltd. Reading, UK (2006), 318-325
4. Casati, F., Fugini, M., Mirbel, I.: An Environment for designing Exceptions in Workflows. Information Systems Vol. 24, No.3 (1999) 255-273
5. Marir, F., Watson, I.D.: Representing and Indexing Building Refurbishment Cases for Multiple Retrieval of Adaptable Pieces of Cases. First International Conference on CBR (ICCBR-95), Portugal (1995)
6. van der Aalst, W.M.P, Hofstede, A.H.M., Kiepuzewski B., Barros, A.P.: Workflow Patterns. Distributed and Parallel Databases, 14(3), 5-51 (2002)
7. Russell, N., ter Hofstede, A.H.M., Edmond, D., van der Aalst, W.M.P.: Workflow Data Patterns. QUT Technical report, FIT-TR-2004-01, Queensland University of Technology, Brisbane (2004)
8. Russel, N., van der Aalst, W.M.P, ter Hofstede, A.H.M.,: Workflow Exception Patterns. Proceedings of the 18th International Conference on Advanced Information Systems Engineering (CAiSE 06), Luxembourg (2006)

9. Russel, N., van der Aalst, W.M.P, ter Hofstede, A.H.M.,: Exception Handling Patterns in Process-Aware Information Systems. BPM Center Report BPM 06-04, BPMcenter.org, Queensland University of Technology, Brisbane (2006)

10. Mulyar, N.A., van der Aaslt, W.M.P.: Patterns in Coloured Petri Nets, Department of Technology Management, Eindhoven University of Technology, The Netherlands, BPM Center Report BPM-05-11, BPMcenter.org (2005)

11. Agostini, A., De Michells, G., Grasso, M. and Patriarca, S. (1993) 'Reengineering process with an innovative workflow management system: a case study.' *Proceedings of the conference on organisational computing systems (COOCS),* Malpitas, California, 01-04 November, pp. 154-165.

12. Dellen, B., Maurer, F. and Pews G. (1997) 'Knowledge-based techniques to increase the flexibility of workflow management.' *Data & Knowledge Engineering,* 23(3), pp. 269-295.

13. Sutcliffe, A.G. and Maiden, N.A.M. (1993) 'Use of Domain Knowledge for RequirementsValidation.' *Proceedings of IFIP WG8.1 Conference on Information System DevelopmentProcess,* 1-3 September, Como, Italy: Elsevier North-Holland.

14. Van der Aalst, W. M. P. (1999) 'Process-oriented architectures for electronic commerce and inter-organisational workflow.' *Information Systems,* 24 (8), pp. 639-671.

15. Van der Aalst, W.M.P., Reijers, H.A., Weijters, A.J.M.M., Van Dongen, B.F., Alves deMedeiros, A.K., Song, M. and Verbeek, H.M.W. (2007) 'Business processmining:application.'*Information Systems,* 32(5), pp.713–732.

16. Jarvis, P., Moore, J., Chung, P., McBriar, I., Stader, J., Ravinranathan, M. and Macintosh, A. (2000) *Applying intelligent workflow management in the chemicals industries*. Published in Association with the Workflow Management Coalition.

17. Cardoso, J. & Sheth, A. (2002) *Semantic e-workflow composition.* Technical Report No. 02-004, LSDIS Lab, Computer Science Department, University of Georgia, Athens GA.

18. Zhuge, H., Chen, J., Feng, Y. and Shi, X. (2002) A federation-agent-workflow simulation framework for virtual organisation development, *Information & Management,* 39(4), pp. 325-336.

19. Chung, L.P.W., Cheung, H., Stader, J., Jarvis, P., Moore, J. and Macintosh, A. (2003) 'Knowledge-based process management - an approach to handling adaptive workflow.' *Knowledge-Based Systems, 16, pp. 149-160.*

20. Kaster, D.S., Medeiros, B.C. and Rocha, V.H. (2005) 'Supporting modelling and problem solving from precedent experiences: the role of workflows and case-based reasoning.' *Environmental Modelling & Software,* 20, pp. 689-704.

21. Cardoso, J., Bostrom, R. P., & Sheth, A. P. (2004). Workflow Management Systems and ERP Systems: Differences, Commonalities, and Applications. *Information Technology and Management, 5 (3-4), 319-338.*

22. Hollingsworth, D. (1994) The Workflow Reference Model: *The Workflow management Coalition Specification, Document Number TC00-1003, Document Status-Issue 1.1*