

See discussions, stats, and author profiles for this publication at: <http://www.researchgate.net/publication/228783438>

Flexible workflow execution through an ontology-based approach

CONFERENCE PAPER · OCTOBER 2004

CITATIONS

4

READS

11

3 AUTHORS, INCLUDING:



[Marco A. Casanova](#)

Pontifícia Universidade Católica do Rio de Ja...

254 PUBLICATIONS 1,794 CITATIONS

SEE PROFILE

Flexible Workflow Execution through an Ontology-based Approach

Tatiana Almeida S. C. Vieira

Marco Antonio Casanova

Department of Informatics, Pontifical Catholic University of Rio de Janeiro
Rua Marquês de São Vicente, 225 - CEP 22.453-900 - Rio de Janeiro, RJ - Brazil
{tati,casanova}@inf.puc-rio.br

Abstract

Workflow execution is many times stopped or cancelled due to unavailability of resources or lack of information. However, recent workflow applications usually require more relaxed execution. To flexibilize workflow execution, two mechanisms are proposed in this paper: a mechanism to handle presuppositions, which allows execution to proceed in the presence of incomplete information, by adopting presuppositions; and a mechanism for choosing alternatives, used when the workflow definition is abstract or in the presence of negative information. These two mechanisms are based on an ontology, called here workflow ontology, which represents relationships between workflows, resources and users.

1. Introduction

Workflow management systems (WfMSs) received considerable attention lately, motivated by their wide spectrum of applications. Standardization efforts are under way in the context of consortiums, such as WfMC (*Workflow Management Coalition*). Other consortiums, such as W3C, are not directly involved with WfMSs, but they contribute to their evolution by defining protocols and workflow language standards.

Requirements for WfMSs comprise a long list, among which we may highlight distributed execution, cooperation, coordination, and synchronization, that model the way user communities work cooperatively to perform a given task [11] [1].

This paper addresses a complementary family of requirements, that we collectively call *flexible execution*. Briefly, workflow management systems usually interpret a workflow definition rigidly in the sense that the system follows precisely the order of the tasks the definition imposes. However, there are real life situations where users should be allowed to deviate from the prescribed sequence of acti-

ons for various reasons, including unavailability of the required resources.

The so-called *workflow adaptative systems* were proposed as an alternative to achieve flexible execution. Briefly, such systems allow the workflow structure to be changed at runtime, without necessarily cancelling the workflow execution. The works reported in [17] [14] [3] [12] [13] [15] [9] [10] exemplify this approach and are related to the results reported in this paper.

To achieve flexible execution, we propose in this paper: (i) a *mechanism to handle presuppositions* that allows execution to proceed in the presence of incomplete information; and (ii) a *mechanism for choosing alternatives* to subworkflows, resources and users, thereby allowing workflow execution to proceed when the predefined subworkflow or resource, or the pre-assigned user is unavailable (i.e., in the presence of negative information). In some sense, these mechanisms apply the idea of component substitution to the case of workflow execution, where subworkflows, resources and users play the role of components [2].

It must be noted that our flexibilization mechanisms are not intended to allow users to directly interfere with workflow execution. These mechanisms rather use previous semantic information about workflows, their resources and the available users, modelled as a workflow ontology, to suggest better alternatives to the user, or to make educated guesses about missing data, that allow execution to proceed when otherwise it would have been stopped.

The two flexibilization mechanisms are mainly based on:

- the *workflow* the user submitted, that defines relationships between tasks, resources and users;
- a *workflow ontology*, that models semantic relationships between objects (workflows, resources and users), and that covers an application area;
- *semantic rules*, that dictates the way alternatives can be found to allow workflow execution to continue.

The paper is organized as follows. Section 2 presents the example that will be used throughout the paper. Section 3 shows the workflow ontology used by the flexibili-

zation mechanisms proposed. Section 4 describes the flexibilization mechanisms. Finally, Section 5 contains the conclusions of this work.

2. A Brief Motivating Example

The example presented in this section is inspired on a real emergency plan, defined in a large (paper) document. The plan has been translated into a workflow that runs under the InfoPAE system [6]. Albeit very simplified and schematic, the example retains the essential characteristics of the original plan.

Consider the problem of cleaning coastal areas affected by an oil spill. This problem is addressed by defining a set of cleaning procedures that take into account the oil type (Type I through Type V) and the coastal area characteristics (in this case, Sand beach), as showed in Table 1. Cells are filled with a weight indicating the environmental impact of each of the procedures: 0.00 indicates the smallest environmental impact, 0.25 some impact, 0.50 a significant impact, 0.75 the greatest impact, and 1.00 inapplicable.

Now, suppose that the user comes to a point in the overall emergency plan execution where he needs to select a cleaning procedure for a sand beach affected by an oil spill. The user (or the workflow management system) can then look up in Table 1 to select (or invoke) the best procedure to clean the beach. For example, if the oil is of Type II, the best procedures are “VC: Vacuum Cleaning” and “CL: Cold, Low Pressure Cleaning”.

However, if he has no information about the oil type, Table 1 becomes useless. In this case, he may take an educated guess and assume, say, that the oil is of Type II. He will then proceed with the emergency plan based on this assumption. Moreover, if he cannot execute the best procedures for Type II oil (those with weight 0.00), because a predefined resource is unavailable, then he may resort to “UA: Use of Absorbents” and “CH: Cold, High Pressure Cleaning”, which are the second best choices (those with weight 0.25).

3. Workflow Ontology

A *workflow ontology* is at the core of the flexibilization mechanisms proposed here. See Figure 1 for a simplified RDF [16] graph, and access <http://www.inf.puc-rio.br/~tati/ontologies/workflow.owl> and <http://www.inf.puc-rio.br/~tati/ontologies/br.2.owl> for the workflow ontology and an example written in OWL, respectively.

Basically, a workflow ontology represents relationships between abstract and concrete workflows, resources and users. Also, this ontology indicates which resources and

users are required to execute each workflow. The relationship between abstract and concrete objects is explored by the mechanism for choosing alternatives, as explained later.

This ontology, in addition to other information, can be seen as a formal and semantic way to represent the table presented in the last section of this paper.

This ontology guides the discovery of possible alternatives when the execution of a workflow instance fails to proceed. Based on the subworkflow that caused the execution to halt and on the execution context, the system traverses the ontology to discover a suitable replacement for the subworkflow that failed to execute (or to discover an alternative resource or user that will let the halted subworkflow to proceed).

4. Flexibilization Mechanisms

This section describes two mechanisms that allow workflow execution to proceed in the presence of incomplete information, by adopting presuppositions, and in the presence of negative information (or of an abstract definition), by suggesting alternatives to the execution. The idea behind the flexibilization mechanisms proposed in this paper is very similar to the query relaxation strategy used in the CoBase system [7] and in the CoSent system [8].

The two flexibilization mechanisms use workflow ontologies, in addition to:

- *presupposition rules*, to compute presuppositions, when the required information is unavailable. These rules can state, for instance, to always select oil *Type II* as the default value for oil type;
- *consistency rules*, to assess the presuppositions made when workflow execution terminates. For instance, if the default oil type is II, the consistency rule must define that a computation is valid iff, during the computation, if the value of oil type became known after the presupposition rule was used, then the known oil type must be Type II;
- *semantic proximity properties*, that help finding alternative subworkflows, resources and users, when the predefined subworkflow or resource, or the pre-assigned user is unavailable. These properties give the strength of the relationship between the various classes of objects: workflows, resources and users;
- *semantic rules*, that dictated the way alternatives can be found, for instance, based on the last parallel workflow executed and on the execution context. These rules basically define the semantic proximity properties for each execution context situation.

Table 1. Environmental impact of cleaning procedures for sand beaches.

Oil Type	Type I	Type II	Type III	Type IV	Type V
ND: Natural Degradation	0.00	1.00	1.00	1.00	1.00
UA: Use of Absorbents	1.00	0.25	0.00	0.00	0.00
VC: Vacuum Cleaning	1.00	0.00	0.00	0.00	0.00
CL: Cold, Low Pressure Cleaning	1.00	0.00	0.00	0.25	0.25
CH: Cold, High Pressure Cleaning	1.00	0.25	0.25	0.25	0.25
HL: Hot, Low Pressure Cleaning	1.00	1.00	0.50	0.50	0.50
HH: Hot, High Pressure Cleaning	1.00	1.00	0.50	0.50	0.50
PC: Vapor Cleaning	1.00	1.00	0.75	0.75	0.75

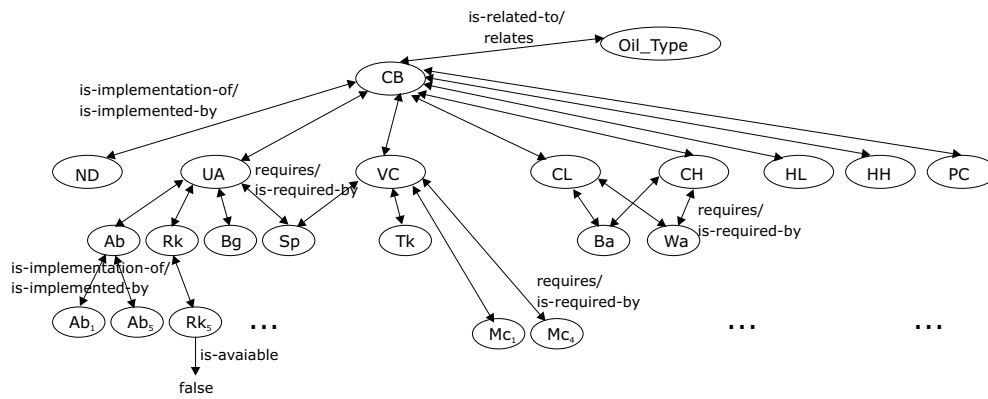


Figure 1. Fragment of a workflow ontology that shows the beach cleaning procedures (refer to Table 1 for the abbreviations used).

The mechanism to handle presuppositions is invoked whenever the required information is unavailable and essentially interprets the presupposition rules. It may be invoked in two distinct points of the execution. First, when delegating a subworkflow instance W' of a workflow instance W to another user U' , the coordinator U of W may find out that the value of a variable V , required to test the pre-condition C' of W' , is missing. In this case, U invokes the presupposition mechanism to find out a value p for V , tests C' with p , and delegates W' to U' , if C' succeeds when V receives p as value. Second, when is executing W , he may also miss the value of some variable T . Then, U may again invoke the presupposition mechanism to find out a value for T .

The mechanism for choosing alternatives operates in two different modes, which are essentially equivalent, but require slightly different interpretations (and modelling) of the semantic proximity properties.

In the first mode, the workflow definition contains specific subworkflow, resources and users, but it may indicate that the assignment is flexible. In this case, the mechanism

is invoked when the predefined subworkflow or resource, or the pre-assigned user is unavailable (i.e., in the presence of negative information). It uses the semantic proximity properties to:

- find a semantically equivalent subworkflow such that the required resources and the assigned user are available;
- find semantically equivalent resources that are available;
- find a user, with equivalent capacities, that is available.

It is worth noting that these three choices are mutually dependent. For instance, if an alternative subworkflow is considered, one must verify if the required resources and users are available. Likewise, if an alternative user is considered, one must check if he can really execute that subworkflow and if the resources are available.

In the second operation mode, the workflow definition specifies an abstract subworkflow or task, leaving it to the mechanism, or in some extent to the user itself, to decide,

at run time, which is the best alternative that can be executed, that is, whose resources and users are all available.

Moreover, in both modes, typically more than one alternative is possible. Hence, the semantic proximity properties must provide some form of cost estimate, and the mechanism for choosing alternatives must be equipped with a cost model that is invoked to select the best alternative, or at least to guide some heuristics that will select a reasonable alternative. The alternatives must be chosen considering also the availability of users and resources, as indicated by the *is-available* property of the workflow ontology.

When a subworkflow instance terminates, the workflow management system must invoke the consistency rules to analyze the results produced, if presuppositions were made.

Besides the abstract modelling of subworkflows, resources or users, from the semantic point of view, there is another level of abstraction, concerned with technology independence. The workflow and the respective workflow ontology are defined in any high level language, and a *binding* is made, at execution time, to the chosen technology. For instance, each workflow can be implemented as a Web service, and the architecture can be based on this trend, using WS-Transaction [4] and WS-Coordination [5] frameworks.

5. Conclusion

We presented in this paper two mechanisms to flexibilize the execution of workflow instances: a mechanism to handle presuppositions that allows workflow execution to proceed in the presence of incomplete information, and a mechanism for choosing alternative subworkflows, resources and users in the presence of negative information, or when abstract definitions are adopted.

These mechanisms use additional semantic information about the workflow definitions, resources and users involved, contained in the workflow ontology. This ontology guides the discovery of possible alternatives when the execution of a workflow instance fails to proceed, and helps assigning default values for undefined variables, with the help of additional semantic information.

Acknowledgment

This work was partially supported by CNPq, under grants 140600/01-9 and 55.2040/02-9. We gratefully acknowledge the fruitful discussions with the InfoPAE development team at TeCGraf/PUC-Rio.

References

- [1] Gustavo Alonso, Divyakant Agrawal, Amr El Abbadi, and C. Mohan. Functionality and Limitations of Current Workflow Management Systems. *IEEE Expert*, 2(5), 1997.
- [2] Valeria De Antonellis, Michele Melchiori, and Pierluigi Plebani. An Approach to Web Service Compatibility in Cooperative Processes. In *Proceedings of the Symposium on Applications and the Internet Workshops*, pages 95-100. IEEE, January 2003.
- [3] Iliia Bider and Maxim Khomyakov. Is it Possible to Make Workflow Management Systems Flexible? Dynamical Systems Approach to Business Processes. In *Proceedings of the 6th International Workshop on Groupware (CRIWG 2000)*, pages 138-141, October 2000.
- [4] Felipe Cabrera, George Copeland, Bill Cox, Tom Freund, Johannes Klein, Tony Storey, and Satish Thatte. Web Services Transaction (WS-Transaction). <http://www.ibm.com/developerworks/library/ws-transpec/>, August 2002. IBM, DeveloperWorks.
- [5] Luis Felipe Cabrera, George Copeland, William Cox, Max Feingold, Tom Freund, Jim Johnson, Chris Kaler, Johannes Klein, David Langworthy, Anthony Nadalin, David Orchard, Ian Robinson, John Shewchuk, and Tony Storey. Web Services Coordination (WS-Coordination). <http://www.ibm.com/developerworks/library/ws-coor/>, September 2003. IBM, DeveloperWorks.
- [6] Marco A. Casanova, Marcelo Tílio M. de Carvalho, and Juliana Freire. The Architecture of an Emergency Plan Deployment System. In *III Simpósio Brasileiro de Geoinformática (GeoInfo' 2001)*, Rio de Janeiro, RJ, 2001.
- [7] Wesley W. Chu, Q. Chen, and M. Merzbacher. *Studies in Logic and Computation: Nonstandard Queries and Nonstandard Answers*, volume 3, chapter CoBase: a Cooperative Database System, pages 41-72. Oxford University Press, New York, 1994. Edited by R. Demolombe and T. Imielinski.
- [8] Wesley W. Chu and Wenlei Mao. CoSent: a Cooperative Sentinel for Intelligent Information Systems, March 2000. Computer Science Department - University of California, LA.
- [9] Soon Ae Chun and Vijayalakshmi Atluri. Ontology-based Workflow Change Management for Flexible eGovernment Service Delivery. In *Proceedings of the Third National Conference on Digital Government*, pages 131-134, Boston, MA, USA, May 2003.
- [10] P. W. H. Chung, L. Cheung, J. Stader, P. Jarvis, J. Moore, and A. Macintosh. Knowledge-based Process Management – an Approach to Handling Adaptive Workflow. In *Knowledge-Based Systems*, volume 16, pages 149-160, April 2003.
- [11] Dimitrios Georgakopoulos, Mark F. Hornick, and Amit P. Sheth. An Overview of Workflow Management: from Process Modeling to Workflow Automation Infrastructure. *Distributed and Parallel Databases*, 3(2):119-153, April 1995.
- [12] Daniela Grigori, François Charoy, and Claude Gobart. Flexible Data Management and Execution to Support Cooperative Workflow: the COO Approach. In *Proceedings of the Third International Symposium on Cooperative Database Systems for Advanced Applications (CODAS 2001)*, pages 124-131, April 2001.
- [13] J. J. Halliday, S. K. Shrivastava, and S. M. Wheeler. Flexible Workflow Management in the OPENflow System. In *Proceedings of the Fifth IEEE International Enterprise Distributed Object Computing Conference (EDOC '01)*, pages 82–92. IEEE, September 2001.

- [14] G. Joeris. Defining Flexible Workflow Execution Behaviors. In *Enterprise-wide and Cross-enterprise Workflow Management - Concepts, Systems, Applications, GI Workshop Proceedings - Informatik '99*, pages 49-55, 1999. Ulmer Informatik Berichte Nr. 99-07.
- [15] Peter Mangan and Shazia Sadiq. On Building Workflow Models for Flexible Processes. In *ACM International Conference Proceeding Series - Proceedings of the Thirteenth Australasian Conference on Database Technologies (ADC'2002)*, volume 5, pages 103-109, Melbourne, Victoria, Australia, 2002. Australian Computer Society, Inc. Darlinghurst.
- [16] W3C. RDF Primer. W3C Recommendation, February 2004. <http://www.w3.org/TR/2004/REC-rdf-primer-20040210/>.
- [17] Mathias Weske. Flexible Modeling and Execution of Workflow Activities. In *Proceedings of the Thirty-First Hawaii International Conference on System Sciences*, volume 7, pages 713-722, January 1998.