An XML Representation for Crew Procedures

Final Report
NASA Faculty Fellowship Program - 2004

Johnson Space Center

Prepared by:	Richard C. Simpson, PhD, ATP
Academic Rank:	Assistant Professor
University & Department:	University of Pittsburgh Department of Rehabilitation Science and Technology Pittsburgh, PA 15260

NASA/JSC

Directorate:	Engineering
Division:	Automation Robotics and Simulation
Branch:	Intelligent Systems
Directorate:	Space Operations
Division:	Automation and Robotics
Branch:	Engineering
JSC Colleague:	Cliff Farmer
Date Submitted:	August 6, 2004
Contract Number:	NAG 9-1526 and NNJ04JF93A

ABSTRACT

NASA ensures safe operation of complex systems through the use of formally-documented procedures, which encode the operational knowledge of the system as derived from system experts. Crew members use procedure documentation on the ground for training purposes and on-board space shuttle and space station to guide their activities. Investigators at JSC are developing a new representation for procedures that is *content-based* (as opposed to *display-based*). Instead of specifying how a procedure should look on the printed page, the content-based representation will identify the components of a procedure and (more importantly) how the components are related (e.g., how the activities within a procedure are sequenced; what resources need to be available for each activity). This approach will allow different sets of rules to be created for displaying procedures on a computer screen, on a hand-held personal digital assistant (PDA), verbally, or on a printed page, and will also allow intelligent reasoning processes to automatically interpret and use procedure definitions.

During his NASA fellowship, Dr. Simpson examined how various industries represent procedures (also called *business processes* or *workflows*), in areas such as manufacturing, accounting, shipping, or customer service. A useful method for designing and evaluating workflow representation languages is by determining their ability to encode various *workflow patterns*, which depict abstract relationships between the components of a procedure removed from the context of a specific procedure or industry. Investigators have used this type of analysis to evaluate how well-suited existing workflow representation languages are for various industries based on the workflow patterns that commonly arise across industry-specific procedures. Based on this type of analysis, it is already clear that existing workflow representations capture discrete flow of *control* (i.e., when one activity should start and stop based on when other activities start and stop), but do not capture the flow of data, materials, resources or priorities. Existing workflow representation languages are also limited to representing sequences of discrete activities, and cannot encode procedures involving continuous flow of information or materials between activities.

INTRODUCTION

NASA ensures safe operation of complex systems through the use of formally-documented procedures, which encode the operational knowledge of the system as derived from system experts. Crew members use procedure documentation on the ground for training purposes and on-board space shuttle and space station to guide their activities. NASA is currently moving from a print-oriented PDF representation to an XML representation for procedures, but the XML representation seeks simply to mimic the PDF look and feel without including any semantic or syntactic information. In other words, there is no explicit identification of procedure *components* (e.g., resources, activities, warnings, pre-conditions, post-conditions) or rules about how components interact. This makes it impossible for intelligent reasoning processes to use this representation for tasks like validation and verification, execution tracking and procedure assistance.

As an alternative, investigators at JSC are developing a new representation for procedures that is *content-based* (as opposed to *display-based*). Instead of specifying how a procedure should look on the printed page, the content-based representation will identify the components of a procedure and (more importantly) how the components are related (e.g., how the activities within a procedure are sequenced; what resources need to be available for each activity). This approach will allow different sets of rules to be created for displaying procedures on a computer screen, on a hand-held personal digital assistant (PDA), verbally, or on a printed page, and will also allow intelligent reasoning processes to automatically interpret and use procedure definitions. The initial goal of the project is to develop a content-based representation for procedures that can be used in place of the existing display-based representation. Once the representation has been developed, editing tools will be developed and tested using actual NASA systems, procedures and system experts. Ultimately, the representation will be used by intelligent systems to provide adaptive training, assistance and monitoring.

During his NASA fellowship, Dr. Simpson examined how various industries represent procedures (also called *business processes* or *workflows*), in areas such as manufacturing, accounting, shipping, or customer service. Content-based workflow representations can be displayed graphically or textually, and there is often a direct mapping between a graphical and textual representation of a workflow. Graphical workflow representations (e.g., UML Activity Diagrams, Petri-Nets, Gantt Charts, BPMN [1]) are typically easier for humans to understand and manipulate, but textual representation languages (e.g., XPDL [2], BPML [3]) can be interpreted by *workflow engines* to automatically manage and monitor execution of business processes.

WORKFLOW MANAGEMENT

Workflow management technology is used to automate business processes in which data and tasks are passed between (human and machine) participants according to a defined set of rules to achieve an overall business goal. Workflow management technology is most frequently used in office environments in applications such as accounting, shipping and general administration, but it is also applicable to design, engineering and manufacturing [4]. An emerging use of workflow management technology is within web sites [5], to automate interactions between a user, the website, and the website's underlying business infrastructure.

A workflow management system automates a business process by managing the sequence of work activities and invoking the appropriate human and/or information technology (IT) resources associated with each activity as specified in a process definition [4]. A process definition consists of a network of activities and their relationships, criteria to indicate the start and termination of the process, and information about each activity within the process such as participants, associated IT applications and data [[6]].

The process definition is expressed in a textual or graphical form or in a formal language notation, which we refer to as a workflow representation language [4]. Textual formats work well for linear tasks, but not for tasks with lots of branching. Difficult to get an "overview" of the task. Difficult to express dependencies within task. Graphical formats provide a good overview of a process but the symbols don't provide room for much detail. Graphical formats often don't have a natural way to represent groupings or hierarchies among steps [7].

Each activity within a process is a single logical step in the process (e.g., making a payment, filing an invoice). It is sometimes not practical to automate all activities within a process, but the process definition will still describe all activities whether they are performed automatically or manually. For example, if a document must be signed in front of a witness, then this might be the one manual activity within an otherwise automated process [6].

WORKFLOW REPRESENTATION LANGUAGES

Unified Modeling Language (UML)

UML provides a visual, object oriented (OO) modeling notation that is valuable for designing and understanding complex systems. UML is the most widely known modeling notation, has a graphical notation which is readily understood, and a rich set of semantics for capturing key features of OO systems [8]. Unfortunately, no single type of UML diagram captures all of the information needed to describe a process. UML activity diagrams can represent complicated sequences and parallelism, but are not the best choice for representing the relationships between activities and objects. UML interaction diagrams do a much better job describing how actions and objects collaborate [9].

Petri-Nets

A Petri Net is a particular kind of directed graph with an initial state called initial marking. The underlying graph of a Petri Net is a directed, bipartite graph consisting of two kinds of nodes, called places and transitions. Arcs represent connections between nodes. An arc can only connect from a place to a transition or from a transition to a place. Connections between two nodes that are of the same kind are not allowed. In graphical representation, places are drawn as circles and transitions as bars or boxes. A marking (state) is an assignment of tokens to the places of the Net. A transition is enabled if each place connected to the transition input arc (input place), contains at least one token. The firing of an enabled transition removes a token from each input place and deposits a token on each place connected with its output arcs (output place). At any given time instance, the distribution of tokens on places defined the current state of the Petri Net; thus, the modeled system. Petri Nets also allow the determination of reachability (if a reachable/obtainable from a given state) and deadlocking (if a state could be reached where the process can not proceed) [10].

Business Process Modeling Notation (BPMN) [1]

The Business Process Modeling Notation (BPMN) specification provides a graphical notation for expressing business processes in a Business Process Diagram (BPD). The objective of BPMN is to support business process management by both technical users and business users by providing a notation that is intuitive to business users yet able to represent complex process semantics. The BPMN specification also provides a mapping between the graphics of the notation to the underlying constructs of execution languages, particularly BPEL4WS [1].

Process Specification Language (PSL) [9]

PSL allows for the possibility of multiple syntaxes, with the choice of syntax depending on factors such as the nature of the process being described and the data source and destination. Key to PSL are the formal definitions (ontology) that underlie the language. Because of these explicit and unambiguous definitions, information exchange can be achieved without relying on hidden assumptions or subjective mappings. PSL semantics are represented using a formal language developed for the exchange of knowledge among disparate computer programs. Thus concepts can within a process be defined unambiguously, a necessary characteristic to exchange process information using the PSL ontology [9].

Business Process Modeling Language (BPML) [3]

BPML provides an abstract model for expressing business processes and supporting entities. BPML defines a formal model for expressing abstract and executable processes that address all aspects of enterprise business processes, including activities of varying complexity, transactions and their compensation, data management, concurrency, exception handling and operational semantics. BPML also provides a grammar in the form of an XML Schema for enabling the persistence and interchange of definitions across heterogeneous systems and modeling tools [3].

Business Process Execution Language for Web Services (BPEL4WS) [5]

BPEL provides an XML notation and semantics for specifying business process behavior based on Web Services. A BPEL4WS process is defined in terms of its interactions with partners. A partner may provide services to the process, require services from the process, or participate in a two-way interaction with the process. Thus BPEL orchestrates Web Services by specifying the order in which it is meaningful to call a collection of services, and assigns responsibilities for each of the services to partners [8].

WORKFLOW PATTERNS

A useful method for designing and evaluating workflow representation languages is by determining their ability to encode various *workflow patterns* [11], which depict abstract relationships between the components of a procedure removed from the context of a specific procedure or industry. Investigators have used this type of analysis [10-12] to evaluate how well-suited existing workflow representation languages are for various industries based on the workflow patterns that commonly arise across industry-specific procedures.

Based on this type of analysis, it is already clear that existing workflow representations capture discrete flow of *control* (i.e., when one activity should start and stop based on when other activities start and stop), but do not capture the flow of data, materials, resources or priorities. Existing workflow representation languages are also limited to representing sequences of discrete activities, and cannot encode procedures involving continuous flow of information or materials between activities.

USING XML AS THE BASIS FOR A WORKFLOW REPRESENTATION LANGUAGE

An XML markup scheme for process data should take advantage of what XML does best, while minimizing the impact of where XML falls short. XML's "tag-centric" syntax makes it a natural fit for representing ordered sequences and hierarchies. Thus it is well suited for ordering time points and occurrences of activities. It is also good at representing sub-activities and sub-occurrences. Another capability of XML, useful for process representation, is XML's modularity. For example, using XML namespaces I can embed an arbitrary object description into a process specification and leave it up to a software tool, separate from the process specification interpreter, to parse the object description. I can also employ namespaces to modularize our process markup language itself (perhaps mirroring PSL's modularization) [9].

Although XML has many advantages for representing processes, it has a major disadvantage. While XML excels as a serialization syntax for exchanging data structures between applications, XML is not very good at expressing the kinds of complex constraints needed for process descriptions. For example, it might be difficult for an

XML schema for a process description language to enforce scheduling constraints involving shared resources. Such constraints could be more easily expressed in a rich language for knowledge representation such as KIF [9].

Because XML is deficient when it comes to representing complex constraints on populations of data elements, its process representation capabilities are limited. However, this does not mean that I cannot use XML to exchange process descriptions. Rather, it means that I probably would not want to exchange all of a process description's underlying ontology in XML, and I cannot count on an XML language to enforce all constraints on process data. It also means that XML would be a poor authoring environment for all but the most simple process descriptions [9].

REFERENCES

[1] S. A. White, "Business Process Modeling Notation (BPMN)," Business Process Management Initiative (BPMI) 3 May 2004.

[2] WfMC, "Workflow Process Definition Interface -- XML Process Definition Language," Workflow Management Coalition, Lighthouse Point, FL, Workflow Standard WFMC-TC-1025, 25 October 2002.

[3] A. Arkin, "Business Process Modeling Language," Business Process Management Initiative, Specification 13 November 2002.

[4] "Workflow Management Facility Specification, V1.2," Object Management Group, Needham, MA April 2000.

[5] T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, and S. Weerawarana, "Business Process Execution Language for Web Services, v1.1," 5 May 2003.

[6] R. Allen, "Workflow: An introduction," in *The Workflow Handbook 2001*, L. Fischer, Ed. Lighthouse Point, FL: Future Strategies, Inc., 2001, pp. 15-38.

[7] D. R. Wieringa and D. K. Farkas, "Procedure writing across domains: Nuclear power plant procedures and computer documentation," presented at International Conference on Systems Documentation, Chicago, IL, 1991.

[8] K. Mantell, "From UML to BPEL," vol. 2004: IBM developerWorks, 2003.

[9] D. Dodds, A. Watt, M. Birbeck, J. Cousins, D. Rivers-Moore, R. Worden, M. Nic, D. Ayers, K. Ahmed, A. Wrightson, and J. Lubell, *Professional XML Metadata*. Hoboken, NJ: Wrox Press, 2001.

[10] A. Knutilla, C. Schlenofff, S. Ray, S. T. Polyak, A. Tate, S. C. Cheah, and R. C. Anderson, "Process Specification Language: Analysis of Existing Representations," National Institute of Standards and Technology, Gaithersburg, MD NISTIR 6133, 1998.

[11] W. M. P. van der Aalst, A. H. M. ter Hofstede, B. Kiepuszewski, and A. P. Barros, "Workflow Patterns;" *Distributed and Parallel Databases*, vol. 14, pp. 5-51, 2003.

[12] C. Schlenoff, A. Knutilla, and S. Ray, "Unified Process Specification Language: Requirements for Modeling Process," National Institute of Standards and Technology, Gaithersburg, MD NSTIR 5910, September 1996.