Using Process Definitions to Facilitate the Specification of Requirements

Leon J. Osterweil, Norman K. Sondheimer, Lori A. Clarke, Ethan Katsh
University of Massachusetts, Amherst, MA 01003
{ljo, sondheimer, clarke}@cs.umass.edu, katsh@legal.umass.edu
Daniel Rainey
National Mediation Board
Washington, DC
rainey@nmb.gov

Abstract

This paper describes early experiences in using process definitions to facilitate requirements specification. The paper emphasizes addressing the constraints posed by the particularly challenging activity of obtaining requirements for digital government systems. The use of precise definitions of processes is suggested as a vehicle for bringing diverse stakeholders together to agree on system requirements in a transparent environment facilitating trust. A user interface prototype is employed to help make the implications of the process definitions concrete. Iterative incremental application of these technologies is leading to new understandings of how to elicit system requirements in the challenging domain of digital government. Experience from the development of an Online Dispute Resolution system for use by the US National Mediation Board illustrates our approach.

1. Background

The creation of even straightforward digital government applications has lagged parallel developments in e-commerce, due in part to the stringent requirements imposed by democratic systems of government for collaboration among all stakeholders and in part to the need to provide transparency to citizens [1].

To meet the requirements for consultation and collaboration in governmental matters it is necessary to be adequately inclusive in establishing effective contact and participation from all stakeholders, including the general citizenry. But, as stakeholder communities become broader, increasing numbers of these communities are likely to be less familiar with software systems, complicating the job of involving these constituencies effectively. Our expectation, however, is that even stakeholders having relatively little familiarity with software systems will still be comfortable and effective in debating the specifics and details of the processes that impinge upon them and their interests. As Jane Fountain has observed, "at organizational and interorganizational levels, actors design and use processes and systems to codify and structure information in order to routinize repeated behaviors, transactions and information processing sequences." [2] Accordingly we suggest that the development of digital government systems should start with the design, analysis, implementation, execution, and modification of government processes, and that these processes then be used as the framework within which the requirements for specific software systems be evolved.

This process-centric view of the world seems to be a promising approach in the development of effective systems in such diverse areas as e-commerce, medical practice [3], and engineering design [4]. In this paper we describe early efforts to adopt this view in digital government application development by merging process definition and prototyping approaches into existing participatory requirements elicitation and analysis activities. Our expectation is that this should lead to the creation of digital government systems that are based upon input from stakeholder constituencies that are sufficiently broad and diverse to meet the needs of digital government.

We note that prior work on approaches such as Joint Application Development (JAD) [5] have also emphasized the importance of involving broad and diverse communities in requirements definition. And, indeed, we also note that the use of user modeling and scenarios have also been advocated and employed as well. More recently, use cases [6] have been advocated as a requirements elicitation approach, and their incorporation into formalisms such as UML have done much to popularize their use [7]. These approaches seem to us increasingly to recognize the importance of understanding the processes within which software systems are to be embedded. Our

approach suggests going farther and using a process language with broad and rigorous semantics to define that process. We suggest that process definition rigor should lead more expeditiously to success in defining software system requirements. Moreover, a sufficiently clear process definition language should also successfully engage wider stakeholder constituencies. Finally, we suggest that rigorous process definitions can be expected to be more effective in the precise evaluation of the effectiveness of changes in continuous software system improvement cycles.

The specific problem domain that we use to demonstrate our approach is the domain of labor-management dispute resolution. This domain impinges upon a very broad spectrum of stakeholder constituencies that vary widely in their contact with, and appreciation for, software systems. In addition, process transparency is particularly effective in projecting the appearance of fairness that seems to be particularly crucial in this domain. Among the more promising approaches to dispute resolution are those that use computer and communication technologies, referred to as Online Dispute Resolution (ODR) [8]. We will describe experience with the U.S. Government's National Mediation Board (NMB) in applying process technology to the development of an ODR system.

2. Approach

A focus on process definition is not unreasonable, as most organizations do have a sense of the importance of process to their work. Unfortunately, however, organizational processes are usually defined either informally, or with diagrams that lack consistency and rigor. Thus, for example, Figure 1 shows NMB's representation of its Interest Based Bargaining process, which is the core of one of its approaches to mediation.
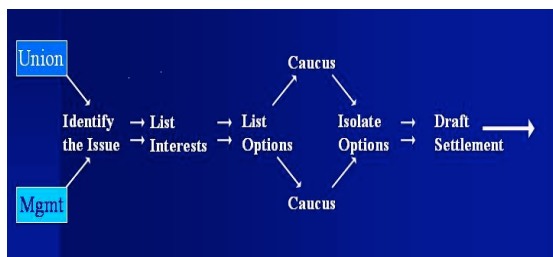

Fig. 1. NMB's Interest-Based Bargaining Process

As can be seen, this representation is at a very high level, and clearly omits many details of the process as actually carried out by NMB's mediators. Such a process representation is inadequate as a basis for determination of the precise requirements for software systems to support it. Our approach dictates the use of a precise language to define this process. Accordingly, we propose the use of a process definition language, such our Little-JIL language [9], to capture the needed details. To illustrate some of the semantic issues, and the degree of rigor, that we believe a process definition language needs to incorporate, here we present only the briefest summary of key Little-JIL features. The interested reader is referred to [10] for more extensive details.

We believe that a process definition language must support the definition of coordination amongst human and automated agents, and at different abstraction levels. It should support the definition of control flow, including the handling of potential exceptional situations, and must also support the definition of how resources are used in a process, and how artifacts flow through the process. The language semantics must be rigorously defined using a mathematical formalism such as predicate logic or finite state machines.

In Little-JIL a process is defined as hierarchy of steps whose semantics are defined in terms of finite state machines. The leaves of this hierarchy represent the smallest specified units of work.

Figure 2 shows the graphical representation of a Little-JIL step with its different badges and possible connections to other steps. The interface badge specifies artifacts that are either required for, or generated by, the step's execution as well as the resources needed to support step execution. Every step specifies an 'agent', either human or automated, that is responsible for executing the step. A step may also include pre- and/or post-requisites, represented by badges on either side of the step. On the left of the black step bar of every non-leaf step, is a control flow badge that specifies the order in which the step's child substeps are to be executed. A child is connected to its parent by an edge that also contains specification of artifact flows between parent and child.
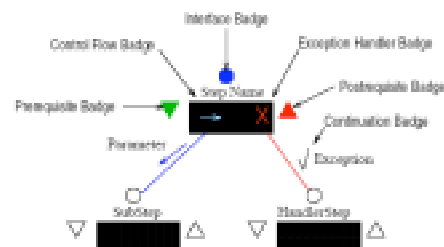

Fig. 2 A Little-JIL step construct

On the right of the step bar is an X sign that represents the exception handling capabilities of the step. Attached to this badge by exception edges are any handlers defined to deal with exceptions that may occur in any of the descendants of this step. Each handler is itself a step and is annotated to indicate the type of exception that it handles. Once a handler is

executed, one of four exception continuation semantics defines where process flow will return.

There are four different non-leaf *step kinds*, namely "sequential", "parallel", "try" and "choice". Children of a "sequential" step are executed one after another from left to right. Children of a "parallel" step can be executed in any order, including in parallel. The "choice" and "try" steps offer human agents alternative ways to complete execution of the step.

A complete Little-JIL process definition also contains definitions of *artifacts* and *resources* to complement this coordination definition. *Artifacts* are entities such as data items, files, or access mechanisms that are passed between parent and child steps, much in the way that parameters are passed in a procedure invocation in a standard programming language. Resources are contested entities (e.g. tools, databases, etc.) that are required to enable the execution of the step. Defining processes in Little-JIL is greatly facilitated by the existence of a suite of tools, including the Visual-JIL graphical editor and the Juliette interpreter for Little-JIL processes.

3. NMB Case Study

ODR seems aptly thought of as a family of processes characterized by their attempts to exploit computer-based capabilities to efficiently and effectively help shape and manage the flow of information and communication between the disputants and a third party, while also protecting privacy and assuring security. NMB has found ODR to be useful in carrying out its mandate to mediate disputes in the airline and railroad industries. We suggest that involving broad constituencies in rigorously defining ODR processes, and by implication, the specific software systems they need, should foster broader, more effective participation and thus engender increased feelings of fairness and trust, which are overriding organizational objectives for NMB, and indeed for all branches of all governments.

The goal of our project is eventually to construct ODR software systems that support precisely defined processes that a broad set of stakeholder groups have agreed synergistically integrate the actions of human participants (e.g. disputants and mediators) with automated computer facilities. But before committing to the development of such full-fledged software systems we have begun by creating some crude early prototypes intended to capture a rough sense of precise NMB processes, and to be used to elicit further, more precise and complete details. As noted above, Figure 1 is a diagram that NMB has used to explain its notion of Interest Based Bargaining (IBB) [12], one of NMB's key approaches to dispute resolution. We have adopted it as the basis of our first attempt to define an ODR process.

We began by interviewing four senior NMB mediators in several multi-day interviews. The mediators took our team through the IBB process training that NMB gives to potential labor and management participants in mediation. NMB then walked us through the details of the specific mediation process used in grievance resolution. As seems typical for processes that are either poorly documented or undocumented, NMB found it easiest to explain their processes through stories and simulated sessions. Using Little-JIL as a device, the process definers generalized the experiences into Little-JIL definitions that NMB inspected for accuracy. It seems important to note that, although grievance mediation might, at first glance, seem to be a process that has little structure, our investigations, and Little-JIL's requirements for completeness and precision, demonstrated otherwise. Indeed, our work has ultimately led to the development of an incomplete, preliminary process definition that currently contains over 150 Little-JIL steps. The definitions emphasize the presence of structure and the need for discipline in such aspects as control of information flow. These aspects strongly suggest a basis for precise specification of the requirements for software systems to support this process.

To be specific, we used Little-JIL process definition as the basis for defining software support for NMB's "brainstorming" process. NMB's grievance mediation process emphasizes the central importance of an in-room meeting (although significant meeting preparation and final written agreement preparation take place outside the meeting room). The process executed at the in-room meeting is referred to by NMB mediators as "brainstorming". The "technology" used during brainstorming consists of a pen and a paper flip chart that serves as the data storage and common work space. NMB suggested that replacing the paper would be an appropriate goal for initial ODR software system support. Thus, our first efforts were aimed at carefully defining the process of brainstorming. Figure 3 depicts a key portion of the Little-JIL definition of the mediator's activities in the brainstorming process. Note that the mediator must execute in parallel (see the "List Items" step) both active participation and control of mediation. There are precise times at which input is solicited from the participants, and other times when summarization must occur. Questions, issues, and comments are to be recorded, and the mediator has the responsibility for deciding which inputs are to be recorded. Note how the Little-JIL process definition makes clear how and
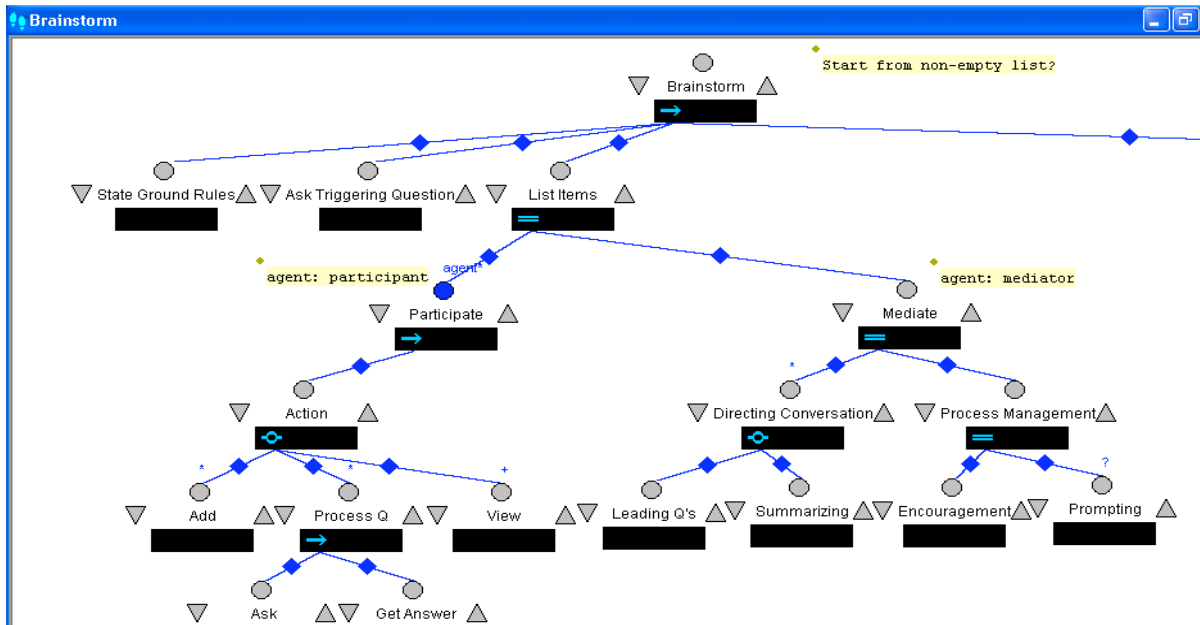
Fig. 3 A small part of the Little-JIL definiton of NMB's Interest-Based Bargaining Process

where these various activities occur. This suggests ways in which software technologies can help support this process, and indicates how our use of process definition strongly facilitates specification of requirements for ODR software.

Based upon the strong suggestions made by the process definition, we next developed a prototype system intended to project an impression of how a fully process-driven ODR grievance mediation support system might appear to the mediation participants. While our eventual goal is to use our Little-JIL process definition execution system (called Juliette) to serve as an interpreter of a fully comprehensive and broad negotiation process definition, we wanted to be sure that the process definition being developed was indeed consistent with the views and expectations of real mediators. Earlier experience with the elicitation of process definitions had shown us that process performers often do not fully appreciate the impact of process details until they obtain some experience as participants in a process-driven system. Thus, our prototype was developed to convey just that sense of what a process-driven system would look and feel like to participants.

The prototype system, called Storm, was developed very rapidly using the Tapestry system [13] for facilitating development of web-based systems. Storm conveys to mediators and disputants a sense of how communications are to be driven by the process, and how various segments of the datasets (e.g. lists of preferred alternatives, comments from other

participants, etc.) are to appear and are to be maintained. While Storm, as a simulation, lacked much of the restrictive and enforcement power desired in a final process-driven system, Storm enabled these disciplines to be administered informally through edicts and strictures laid down by a skilled mediator.

An immediate result of using this prototype was the recognition of important differences between face-to-face negotiation and ODR negotiations as they might be supported by technology. ODR can allow the "pen" to be taken out of the hand of the NMB mediator and can allow access to the data store to be given directly to all participants. With ODR, participants can be permitted simultaneously to submit ideas, questions or clarifications, depending on the phase of the process. With ODR, participants can be granted anonymity. Indeed, with ODR input can be submitted from any Internet accessible workstation and participants might come and go from the activity as they wish. It seems clear that an extensive and careful program of experimentation is required in order to decide just which variations upon the face-to-face process are likely to be most desirable as the ODR processes that NMB will actually use. We expected that a process definition language should be easily able to define these process variations, and found that this was indeed the case for Little-JIL. For example, the brainstorming step representing idea acceptance was initially defined as a sequential step executed by the mediator in the face-to-face process. In the ODR version, it needs only to be replaced with a parallel

step, executable by any participant, in order to model the acceptance of ideas arriving at the same time. The ability precisely to define process variants also enables the specification of processes that differ only in one specific detail, thereby creating the basis for controlled experiments aimed at evaluating such details individually. Thus, this approach also provides a superior tool for careful experimentation. It assures that experimental results are correctly attributed to precise process variations.

Little-JIL and Storm have been used in several exercises aimed at evaluating underlying ODR requirements. Three classes at the University of Massachusetts used Storm in hypothetical grievances from the NMB training set. Half of each class was assigned to act as a team representing one of the two parties in a grievance. Each class was facilitated by a professional mediator. In addition, NMB professional mediators have used one of our ODR simulations to reenact prior cases. After each exercise, all participants completed an evaluation form.

4. Results, Conclusions, and Future Directions

While these early uses of Little-JIL and Storm can not be considered to be definitive, the reactions of early users are encouraging. Our early assessments have focused on determining whether participants were "heard" by the mediator and the other party. This is a significant area of assessment because a basic goal of mediation is to ensure that the parties are able to communicate their interests and ideas to each other in an atmosphere where rancor does not preclude the parties from hearing and understanding each others' positions.

In exercises to date, 83 percent of the participants indicated that they felt heard, and 40 percent indicated that they very much felt heard, by the mediator. In addition, 89 percent of the participants indicated that they felt heard, and 40 percent indicated that they very much felt heard, by the other party. These results suggest that the process and system are indeed trusted. Iterative changes to precise process details (made possible by precise process definition capabilities) will be made based upon feedback from mediators and their clients. They will be evaluated to determine which details have which effects upon user trust. We will, moreover, also evaluate how the involvement of the mediators and their clients in these process iterations has itself led to increased trust.

Response from mediators has been especially gratifying. Mediators have in the past expressed dissatisfaction with the complexity of a previously used third-party system for supporting ODR. Their participation in development of our new ODR system

is beginning with our solicitation of comments about ODR processes. This is beginning with recording of their reactions to the use of Storm, disciplined by mediator enforcement of specific process details, as suggested by their comments and defined using our process language. Thus, for example, we are evaluating the suggestion that all participants be allowed direct access to the store of comments and suggestions. This has dramatically changed the role of the mediator from facilitator controlling the flow of the session to a role that is more equal to that of the other participants. This is reinforced by the ability of parties to enter ideas simultaneously. Little-JIL was used to precisely define a process having these characteristics, and Storm was used to project the effect of executing that process. Early evaluations suggest problems with this. For example, if the mediator wishes to draw attention to a particularly creative new option, he or she will likely not be able to respond before several more ideas have been entered. Anonymity, moreover, has had side effects that were not anticipated. It may contribute to an excessively large number of options entered. The freedom to allow non-face-to-face and non-continuous mediation has also drawn negative comment. In response to these comments we are now working to produce alternative processes that capture the benefits of ODR while addressing these concerns. Storm will be used to project these alternative processes, and the mediators will be engaged in their evaluation.

Although there is no data available yet to illustrate the reaction to our approach by actual labor and management negotiators, a set of parties, after a cursory review of our prototype, have agreed to its use in actual negotiations scheduled to take place in early Spring of 2006.

Thus, we have important evidence that this use of prototypes, disciplined by precisely defined process definitions, is leading to important understandings that seem essential before committing to firm ODR software requirements. We believe, in addition, that the active involvement of the mediators in these considerations will ultimately lead to greater acceptance of the ODR systems that eventually result. Currently, we have compiled a list of over 50 suggested ODR process and system changes and enhancements, addressing issues that range from screen layout and text edit modification to such new functionality as concurrent discussions and support for reorganizing lists of ideas. The majority of suggestions have come from the mediators. We take this to indicate that careful analysis of their process has led them to a clearer understanding of the possibilities of automation.

Our project is continuing to define precise variations of the IBB process that are specialized for different negotiation settings and scenarios. We expect

that these variations will lead to clearer understandings of which exact process features contribute most successfully to effective negotiation in various different settings. This would comprise an important contribution to ODR research.

5. Acknowledgments

6. References

[1]   J. E. Fountain, Information, Institutions and Governance. Cambridge: Harvard University, 2002.

[2]   J. E. Fountain, "Challenges to Organizational Change: Multi-level Integrated Information Structures (MIIS)," in From e-Gov to i-Gov: Governance and Information Technology in the 21st Century, V. Mayer-Schonberger and D. Lazer, Eds., 2006.

[3]   L. A. Clarke, Y. Chen, G. S. Avrunin, B. Chen, R. Cobleigh, K. Frederick, E. A. Henneman, and L. J. Osterweil, "Process Programming to Support Medical Safety: A Case Study on Blood Transfusion," presented at Proceedings of the Software Process Workshop, Beijing, China, 2005.

[4]   N. K. Sondheimer, "A Process Guided e-Design Environment," presented at First Annual International Conference on e-Design, Atlanta, 2005.

[5]   J. Wood and D. Silver, Joint Application Development, Second ed. New York City, New York: Jossey-Bass, 1995.

[6]   I. Jacobson, M. Christerson, P. Jonsson, and G. Overgaard, Object-Oriented Software Engineering: A Use Case Driven Approach. New York: ACM Press, 1992.

[7]   OMG, "Unified Modeling Language (UML) Version 1.4.2." Framingham, MA: Object Management Group, 2001.

[8]   E. Katsh and J. Rifkin, Online Dispute Resolution: Resolving Disputes in Cyberspace. San Francisco: Jossey-Bass, 2001.

[9]   A. G. Cass, B. S. Lerner, E. K. McCall, L. J. Osterweil, S. M. Sutton Jr., and A. Wise, "Little-JIL/Juliette: A Process Definition Language and Interpreter," presented at International Conference on Software Engineering, Limerick, Ireland, 2000.

[10]   A. Wise, "Little-JIL 1.0 Language Report," Department of Computer Science, University of Massachusetts: Amherst, Amherst, MA, Technical Report UM-CS-1998-24, April 1998.

[11]   J. M. Senger, Federal Dispute Resolution: Using ADR With the United State Government. San Francisco: Jossey-Bass, 2004.

[12]   Fisher, Ury, and Patton, Getting to Yes, Second ed: Penquin Books, NY, 1991.

[13]   H. M. L. Ship, Tapestry in Action: Manning, 2004.