alpha-Props: A Rule-Based Approach to 'Active Properties' for Document-Oriented Process Support in Inter-Institutional Environments

Aneliya Todorova and Christoph P. Neumann
Friedrich-Alexander University,
Institute of Computer Science 6 (Data Management)
Erlangen-Nuremberg, Germany
aneliya.todorova@googlemail.com

Abstract: The alpha-Props project aims for the design of a lightweight subsystem, which realizes the active properties in active documents in the context of a distributed and decentralized workflow named alpha-Flow. The alpha-Flow designs a dynamic process evolution within heterogeneous cross-organizational systems. The active documents are used as self-contained artifacts of a document-oriented workflow approach, which combines the content-oriented and activity-oriented workflow paradigms in a hybrid solution. The alpha-Props subsystem is devised to react autonomously to artifact and workflow changes by utilizing a rule engine. The main functionality of the active properties in the alpha-Props system is the altering and distribution of the document they are part of.

1 Introduction & Motivation

Active documents ([DEH+00], [HM00]) are documents which allow direct interaction with themselves. In the context of active documents, *active properties* are outlined as the part of the active document that conduces to the logic and liveliness of a passive document and essentially constructs an active document out of it. Active properties react autonomously to events, trigger actions as a result, or alter their own state. The general idea behind the α-Flow approach [NL10] is to use active documents as a means of dynamically and autonomously steering of activities in medical workflows. The focus of the α-Flow prototype is modeling procedural flows in healthcare, taking place between doctors, insurance companies and other potential parties, in a non-paper, instant, access-controlled, dynamic way, without implying neither the exact number of participants nor the exact steps of the flow to be known in advance.

In the α-Flow approach, artifacts are stored in α-Docs representing a patient's clinical reports. An α-Doc incarnates an α-Episode as in the classical sense of a treatment episode *per* patient. Throughout an episode, many α-Cards are created [NL09]. α-Cards can be of two types: coordination or content. An α-Card consists

of a payload and of an α-Descriptor. The α-Descriptor wraps meta-information about the passive document (a.k.a *adornments*), which allows its manipulation and is used to conduct actions on the α-Card. The payload of a content card is a passive document, such as a PDF-file, a DICOM-file or the like; whereas the payload of the only two existent coordination cards (TSA[1] and CRA[2]) is internal information in the form of lists, which is needed for the collaboration.

Based on the α-Flow concept, a software component is required: the *α–Props*. It represents a generic realization of the active properties known from active documents by means of a rule-based system. The objectives set for the thesis were to build a system that reacts to events that come from outside the system or are triggered within it, reason on them and induce actions as a response. Further requirements were that the active properties had the ability to manage all (for the contribution of the actions relevant) concerned parties in a network of distributed participants.

2 Material & Methods

A comprehensive understanding of processes in complex but intrinsic fields such as medicine and healthcare inclines to the conclusion that a system solution is needed, that is both transformational and reactive. A transformational workflow system takes input and produces output; a reactive system triggers output, dependent on input and predefined knowledge, based on interaction with the environment [HP85]. Rule-based systems offer such reactive behavior [DKM86], [Jac98].

Therefore, a rule-based system was considered as the implementation unit of active documents. The infrastructure from the Drools Framework [Dro10] is used for most of the core components of the α-Props module. Most of these are the common stipulated components of a Rule Engine as constituted in the JSR-94[3].

The complex infrastructure of the distributed α-Docs and their replicas is to be understood as follows: many participants work on many workplaces, each of which has an exemplar of all relevant α-Docs (*per* desktop, *per* participant). The α-Props subsystem of the α-Doc has to synchronize its distributed replications. If a participant is owner of the α-Doc, i.e. they have instantiated it, then they are the only one who have the Master-copy of it; all other participants, involved in the episode, get an exact replica of it. The distribution of changing actions to all replicas has to ensure a coherent state of the distributed replicas. Master exemplars of the content cards differ from the replicas only if they are just created and still marked as *private* [NL09]. Only when the content card is labeled *public*, it is also seen by the other parties, and thus is exposed as an exact replica of it.

[1] Treatment Structure Artifact
[2] Collaboration Resource Artifact
[3] JSR-94 is the Java Specification Request for a Java Rule Engine API [JSR02]

3 Result

Events are triggered by users, either locally or remote, and are delegated to the α-Props component via an editor or the network. Local events originate in an α-Editor component that allows state changes of the α-Docs for the desktop user of the active document. Remote events are the ones that originate from remote participants and which have been sent from another α-Props instance of a remote α-Doc replica.

The α-Props core is based on the Drools components like the Knowledge Base, the Rule Engine, the Knowledge Agent, the Working Memory, the Agenda, and the Drools Pipeline. The propagation of changes is realized in the form of artificially created internal events. Such technical events are also used to transform a user's action-request into an internal event that can be interpreted by the rule engine. Such a request can be to add a content card, or update an existing one (its payload or adornments). These events trigger actions which alter the internal state of the Working Memory as well as provoke their propagation outward the system.

The α-Props system interprets events, reacts on its own and triggers internal as well as external changes on the current state of the artifacts. The component is lightweight and offers interactive interfaces towards an editor, a local storage manager and for both a *Receiver* and a *Sender*. The latter are responsible for the data transfer in and out the Working Memory of the rule system, which lies in the core of the α-Props. The implemented prototype realizes active documents applicable for the exchange of patient's treatment reports.

4 Discussion

The distributed contribution of the artifacts in the prototype is realized very basically at the moment: via sockets; and under the assumption that all participants are online all the time and the distribution takes place on the broadcast principle, i.e. all get the newest state of the artifacts. It is though possible to exclude any participants, not interested in a particular artifact life-cycle, as well as to add any new participant dynamically to the workflow. An initial approach for versioning of the artifacts is realized as well by modifying an extra set of rules that handles new versions.

Many issues regarding network security, data persistence, data and thread synchronization, participant management as well as a more complex version and variant management are still open. The online semantics of the current prototype is also an issue. Nevertheless the α-Props prototype comply to the basic requirements that will allow for a generic solution for active properties in an environment of distributed active documents.

References

[DEH+00] P. Dourish, W. K. Edwards, J. Howell, A. LaMarca, J. Lamping, K. Petersen, M. Salisbury, D. Terry, and J. Thornton. A programming model for active documents. In *Proc of the 13th annual ACM symposium on User interface software and technology*, pages 41–50. ACM New York, NY, USA, 2000.

[DKM86] Klaus R. Dittrich, Angelika M. Kotz, and Jutta A. Mülle. An event/trigger mechanism to enforce complex consistency constraints in design databases. *SIGMOD Rec.*, 15(3):22–36, 1986.

[Dro10] JBoss Drools Documentation. `http://www.jboss.org/drools/documentation.html`, 2010.

[HM00] E. Heinrich and H.A. Maurer. Active documents: Concept, implementation and applications. *Journal of Universal Computer Science*, 6(12):1197–1202, 2000.

[HP85] D. Harel and A. Pneuli. *On the development of reactive systems.* Microelectronics and Computer, Technology Corporation, 1985.

[Jac98] Peter Jackson. *Introduction to Expert Systems.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1998.

[JSR02] Java Community Process. JSR 94 Java Rule Engine API. `http://www.jcp.org/en/jsr/detail?id=94`, 2002.

[NL09] Christoph P. Neumann and Richard Lenz. alpha-Flow: A Document-based Approach to Inter-Institutional Process Support in Healthcare. In *Proc of the 3rd Int'l Workshop on Process-oriented Information Systems in Healthcare (ProHealth '09) in conjunction with the 7th Int'l Conf on Business Process Management (BPM'09)*, Ulm, Germany, September 2009.

[NL10] Christoph P. Neumann and Richard Lenz. The alpha-Flow Use-Case of Breast Cancer Treatment – Modeling Inter-Institutional Healthcare Workflows by Active Documents. In *Proc of the 8th Int'l Workshop on Agent-based Computing for Enterprise Collaboration (ACEC) at the 19th Int'l Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE 2010)*, Larissa, Greece, June 2010.