Unified Programming Model for Instrument-driven e-Science Workflows?

Beth Plale
Indiana University, USA
+1 (812) 855-4373
plale@cs.indiana.edu

Chathura Herath
Indiana University, USA
cherath@cs.indiana.edu

Rahul Ramachandran
University of Alabama Huntsville USA
+1 (256)-824-5157
ramachandran@itsc.uah.edu

ABSTRACT
The scientific knowledge discovery process has been aided recently by advances in cyberinfrastructure that automate the execution of data retrieval, modeling and analysis tasks typically undertaken during scientific exploration. But these infrastructures lack a generalized programming model for integrating real time data from sensors and instruments into the analysis process. In this ongoing work we examine two approaches to stream processing, a rule system and a query language-based system, and argue that either can be made suitable for our user base with enough hand-written code, but can either become as accepted as workflow systems in the science community? What will that take?

Categories and Subject Descriptors
H.3.3 [**Information Search and Retrieval]** *Information filtering.*

General Terms
Experimentation.

Keywords
Scientific workflows, environmental stream processing.

1. INTRODUCTION
Scientific knowledge discovery is typically a human process where a scientist interacts with the data, computational models and tools in his lab and on the Internet. Workflow systems such as Taverna, Apache ODE, and Kepler are gaining popularity amongst the scientific community through their ability to execute sequences of tasks on behalf of a scientist during scientific investigation. These systems when built as part of a larger cyberinfrastructure framework give the scientist tools to construct task graphs of execution sequences, often by means of a visual interface where the scientist connects task boxes together with arcs representing either control flow or data flow. Unlike business processing workflows, scientific workflows need to expose a high degree of detail and control to the scientist both during configuration and during execution. Workflow customization is highly prevalent – to the extent that workflow runs are hardly comparable.

Workflow languages used in science discovery are Turing complete so in theory can support any logic expressible by a programming language. The execution environments supporting the workflow engines, on the other hand, are subject to constraints on physical resources, and hence in practice the workflow task graphs used in science utilize relatively few of the workflow patterns cataloged in [5]. In addition, and importantly for our discussion, these workflows, which are executed on demand, are executed once.

Into this context is introduced the need for science discovery that is responsive to real time information. How does a scientist use the information from 1000 geo-sensors in Southern California to predict earthquakes with prediction times that are better than real time? Soon small-scale weather radars will be installed on cell phone towers across the country, particularly in tornado or flood alleys, giving remarkable amounts of information about the lower 3km of the atmosphere.

Further, while the needs of one science lab can be supported by an ad hoc solution to ingesting streaming weather data for processing for instance, can this capability easily be extended to share the data with social scientists who might examine fault behavior to better understand the poverty in an area? If we can, through simple programming models and abstractions, make scientific discovery involving real-time data accessible to specialists who share and utilize data across scientific domains, we have brought science one step closer to solving the largest of human problems.

Our intention in our research is to consider events processing in the context of the highly stylized, highly controlled human work processes in which the scientist engages, where the workflow engine is a given but insufficient for stream processing.

Our work is motivated by realistic complex e-Science stream processing and workflow examples, in particular, real-time responsive regional severe storm prediction. We have compared two open-source systems: the Drools rule-based system and the Esper SQL-style language processing system, on their ability to express and carry out complex stream processing. Our experimental comparison shows the systems to be reasonably comparable in that either can be made suitable to the task with enough hand-written software. But the larger question is this: can either become accepted as part of a workflow system driven cyberinfrastructure for the science community? What will that take?

2. THE SCIENCE CHALLENGE
We focus on the science domain of regional severe storm prediction and do so for several reasons. First, the mesoscale meteorology community is particularly forward looking in their adoption of workflow-driven cyberinfrastructure. The Annual Spring Forecast Challenge held in the US judges the effectiveness

of new research weather forecast models. This year it is using the LEAD cyberinfrastructure [1] routinely to run forecasts. Second, the international weather community has a long history of monitoring the atmosphere so software and community structures are well established (i.e., Unidata) to deliver data from instruments to research desktop in a matter of seconds. Finally, our group has a deep understanding of the stream and data mining issues in real time atmospheric instrument data, so the application is a particularly well-suited motivator to better understand the capability of rule and language based systems.

The example that guides our work can be intuitively understood as automating the work of the meteorologist researcher who wants to run a weather forecast model in response to developing regional (mesoscale) weather. The task traditionally involves significant human intervention and is not executed in response to weather conditions as they occur. The goal is to have the system monitor real time radar data over an area of the country, identifying and tracking severe storms as they occur during a specified time period, and triggering a weather forecast model run for each unique storm. False positives, that is, severe storms that are identified as such but are not really severe storms, are costly because each weather forecast model run monopolizes 32 processors in parallel for 30 minutes or more.

More specifically, the raw events that must be processed are binary radar data volume scans that are on average 3 MBytes in size. In our current work we use data from the network of over 130 large-scale Doppler radars. A 1000Km by 1000Km region over the US contains on average 5 to 10 radar stations each producing a volume scan every five to seven minutes.

Events processing must produce a quilt of radar data that covers the geographic region for each volume scan of the radars and then apply data mining filters to identify the unique severe storms, while tracking existing storms and their movements. The first step is to filter out the radar events that fall outside the selected spatial domain. We must then find a covering collection. That is, readings that roughly happen at the same time" must be fused or joined [3]. The data mining examines a volume scan for specific wind velocity signatures called Rankine Vortex signatures. These signatures are then clustered using a Storm Clustering Algorithm [4]. Data mining is processor intense, so utilizes multiple processors in parallel. Results are then compared to historical data to determine new and unique severe storms. A trigger for each of these is generated that is fed to a waiting weather forecast workflow. Use of historical data about continuous event processing in stream processing has been called "spanning applications".

3. DIFFERENT APPROACHES

There are two accepted approaches to event detection: rule based approaches and SQL-based approaches. Our work to date has shown that the two paradigms are comparable from a performance perspective, but have strengths and weaknesses in their programming model and their ability to interact in the larger system in which they must exist.

The stream processing research community and industry have developed a number of systems that demonstrate various advanced capability. Stream processing systems accessed and specified through a declarative query language include Streambase, Corel8, Tibco Complex Events Processing, and Oracle Extreme Transaction Processing in the industry space and Aurora, StreamDB, and Calder in the research space. We have chosen to experiment with the Esper system [2] because it is an open source product with good performance. Rule-based systems specify behavior as a set of rules that are selectively triggered when state conditions are right. We have chosen to experiment with the forward chaining rule engine, Drools [6], an open source solution that is currently undergoing extension by the author to be more responsive to temporal conditions[6]. Rule engines have been applied to business process monitoring (i.e., Rapide) and have been integrated into workflow management systems, see TriGSflow.

4. FAR FROM USER ACCEPTANCE?

There are a number of critical requirements to making workflow orchestration responsive to environmental data. We think events processing approaches can be fruitfully applied in part because of the expressivness of the continuous processing problems. But as we stated earlier, scientists are beginning to accept workflow orchestrated activity. Further, they require rich control over configuring and running moderately complex workflows that are run once to completion. What is the most appropriate programming model to support the underlying rich functionality of continuous stream processing and highly configurable workflow systems?

Our experimental study qualitatively and quantitatively assesses both a rules and a declarative query language approach against a canonical instrument-driven e-Science workflow example for purposes of exposing the shortcomings. These shortcomings may point to opportunities to unify the paradigms into a single programming model.

5. REFERENCES

[1] Droegemeier, K., et al., Service-oriented environments for dynamically interacting with mesoscale weather, *Computing in Science and Engineering*, IEEE Computer Society Press and American Institute of Physics, Vol. 7, No. 6, pp. 12-29, 2005.

[2] Esper http://esper.codehaus.org

[3] Li, X., B. Plale, N. Vijayakumar, R. Ramachandran, S. Graves, and H. Conover. Real-time storm detection and weather forecast activation through data mining and events processing. *To appear Earth Science Informatics*, H.A. Babaie, Ed., Springer 2008.

[4] Rushing, B.J., R. Ramachandran, U. Nair, S. Graves, R. Welch, and H. Lin. ADaM: a data mining toolkit for scientists and engineers. *Computers and Geosciences*, 31:607–618, June 2005.

[5] Russell, N., A. H. M. t. Hofstede, W. M. P. v. d. Aalst, and N. Mulyar, Workflow Control-Flow Patterns: A Revised View, *BPM Center Report BPM-06-22*, 2006.

[6] Walzer, K., A. Schill, and A. Löser, Temporal constraints for rule-based event processing. *In Proceedings of the ACM First Ph.D. Workshop in CIKM*, ACM, New York, NY, 93-100, 2007.