Recording the Context of Action for Process Documentation

Ian Wootten and Omer Rana

School of Computer Science, Cardiff University, UK

Abstract. In reviewing evidence about real world processes, being aware of the context in which activities within such processes are performed enables us to make more informed judgements. It is necessary to distinguish between the environment in which a process occurs, and the sequence of activities which form part of the description of that process. Each of these types of information is complementary to understanding the other and therefore making associations between them is also important. Our work has been exploring the use of *context* whilst documenting a process and working toward a solution which incorporates the two. We present an approach to automatically relating properties of workflow actors to the documentation of the process within which these actors are involved.

1 Introduction

Context plays a crucial role in support of evidence for a given argument. Statements which are taken 'out of context' could face criticism from those who note such omissions as a distortion of the original intended meaning. There are a number of definitions of context in distributed systems – Brown [1] defines context to be the elements of a user's environment which the computer knows about. Dey and Abowd [2], refer to context as "any information that can be used to characterize the situation of entities (i.e., whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves". Therefore the amount of information we have about context affects how we interpret the event we have observed. Similarly, the way in which we interpret the recording of some data may be altered based on the context in which it has been generated.

We use Groth et al.'s view on provenance [3] as the process which led to a piece of data, and that such processes may be described using evidence represented in the form of process documentation. In service oriented architectures we believe that the context of an action or sequence of actions may be provided by each actor involved in execution of a process. Records of context may exist outside of the notion of a process or the control of a client and this makes later interpretation based on review of both difficult. Our work has been exploring the use of context in the documentation and structuring of such evidence. Known relationships between a process and its context will be different depending on the application, and it may not always be possible to document both in open,

J. Freire, D. Koop, and L. Moreau (Eds.): IPAW 2008, LNCS 5272, pp. 45–53, 2008.

loosely coupled architectures. In this paper we present a system which documents both the sequence of actions which describe a process and the situation in which those actions took place, enabling both navigation of process documentation and prediction of future actor properties. The rest of this paper is organised as follows: in section 2 we review the relevant literature in the area of context as relating to provenance systems and describe our motivation. In section 3 we describe our model of the context of actions upon an actor, followed by a description of the architecture we have adopted in section 4 and a demonstration of its implementation in section 5. Finally in section 6 we conclude.

2 Background and Motivation

Provenance is important for scientists to be able to record information in order to, for example, ensure experiments are performed correctly and to be able to repeat processes which produced interesting results. As many disparate activities may be involved in such a process, without such recording it is difficult to determine precisely how results have been reached. Several solutions have been developed to capture provenance or enable applications to be *provenance aware*, e.g. in Bioinformatics[11] or Chemical Sciences[9]. The Oxford English dictionary describes context as: *the circumstances that form the setting for an event, statement or idea.* For actors in a service oriented system, documenting a process involves describing each of those steps which comprise it. Research to date has widely addressed documenting messages which are sent between actors[3] as these typically indicate invocation of some functionality. Other actions commonly include the actions of scientists who control such systems, which may be documented in a more ad-hoc manner. Recording single events does not however describe properties or conditions which hold true for the actor over a given period of time. We refer to such properties as a description of the *context* of the action, where the action is the function being performed by a particular entity. A universal agreement on the content of process documentation representing context submitted by actors has not yet been reached. Attempts to provide a generic schema for what is to be recorded as context have so far proved to be fruitless, with the Grid Provenance project[1] choosing not to adopt any formal structure for the state of actors during a process. This is due to the diversity in the types of use cases that are required to be satisfied for all those domains which have unique provenance requirements. Such diversity has led to scientists building a variety of tools able to capture specific contextual data. Recently, the Open Provenance Model has been developed to enable sharing of provenance data amongst different systems that adopt it [8]. In this model, data representing context could be considered *artifacts* as they are immutable pieces of state. As yet, no systems are known to have implemented the model and no formal representation of the model has been specified, but work is ongoing. The model we present focuses less on what the content of this contextual data may be and

[1] http://www.gridprovenance.org

more on how those elements are represented and recorded over time, to be of use during queries of process documentation.

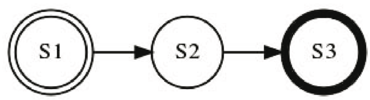
3 Modeling the Context of a Process

We record *process documentation* about past processes to understand what has occurred following invocation of a workflow [3]. In order to document the context of these processes, a set of variables associated with each actor is recorded during workflow execution. The value taken by these variables at any time constitutes the *state* of that actor. It is possible for an actor to progress through a number of different states over a period of time, indicated by a difference in the value of any one of the states variables. When a state changes for an actor, a *state transition* is said to occur from the previous to following state. We assume when state transitions are documented that an actor is only involved in one process at a time.

An example of deriving a number of states from a given set of variables (v_1, v_2, v_3) is given in figure 1(a) with a finite state machine representing the states mined from the data shown in figure 1(b). Here we see that although some common values exist for each variable (such as the value of v_3) a change in any variable can lead to a state change.

State	Observation Time	v_1	v_2	v_3
s_1	1164277522	4.71	13084	2.56
s_2	1164282522	4.71	15698	2.56
s_3	1164287522	4.00	15698	2.56

(a) Deriving unique states from variables

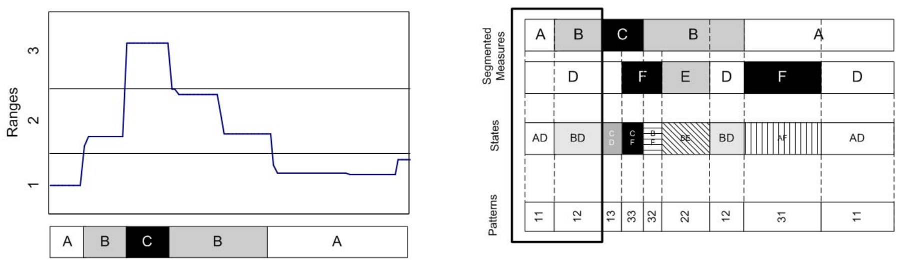


(b) FSM of variable data

Fig. 1.

In the scenario of a service based architecture, the most interesting states are those which occur within the interval when a request message was sent to a service and the associated reply. In this period, the actor's observed state may be documented as a part of any process documentation that is recorded for a process. State is documented using details of the time intervals over which that state applies (i.e start and end times). This means it is possible to mine series which describe the same property as being true over a number of non-overlapping intervals. In figure 2(a) we demonstrate how thresholds are used to segment the measurements of a variable[4], to determine when it was in one of a number of pre-defined ranges. Each of these properties which hold true for a state is documented within a *pattern*. The pattern is an array of values which has the same number of elements as variables that are measured, with each individual element holding a reference to the range which applies. By looking at two series segmented from variables, (as in figure 2(b)) we are able to determine the unique patterns (and hence states) and the times over which they apply. As an example, consider the first highlighted section of figure 2(b). The first two rows correspond to segmented variable measurements and the third indicates over which intervals various combinations of the segmented series applied. Our resultant (state) series therefore describes the periods over which each variable is described as high, medium or low. So for the example, the state series tells us both variables were initially low, whilst a short while later the first variable changed to be within the medium range. The final row indicates the patterns for each state (which correspond to the original ranges), with the upper value indicating the range applying for the series in 2(a).

States observed upon an actor are assumed to be an effect of the request (cause) event from which the observation interval begins. This is in order to support prediction of future actor properties - which would be impossible without causal knowledge.



(a) Interval series mining

(b) Coincidence of interval series

Fig. 2. Mining unique context series from numeric time series

Particular focus has previously been paid to documenting the set of events which comprise a process, without discussion of whether any one of these events may hold causal relationships with properties and conditions holding true for longer than a single point in time. By adopting a representation capable of recording intervals we hope to capture such knowledge.

4 Documenting Context in Service Based Architectures

We use the PreServ software created at the University of Southampton to capture assertions of provenance to a repository known as a *provenance store* [3]. This software has been built in response to a large variety of requirements gathered from numerous domains such as bioinformatics, high energy physics and medicine [5]. PreServ breaks up process documentation into three sub-categories of assertion known as *p-assertions*. Interaction p-assertions document message exchange between services, relationship p-assertions document the causal dependencies between events or data items and actor state p-assertions document the state an actor is in at a given point in time during a process. Dividing documentation into these three types means that parts of it may be recorded by each of the actors which were involved in a process to a repository common to all. PreServ is suitable as a capture mechanism for assertions of state as it does not prescribe their contents, instead leaving it up to specific applications to define this. This leaves us free to specify our own XML representation of state and assert this to storage.

Our implementation of a system capable of automatically documenting state makes use of a State Assertion Registry (StAR) co-located with a service [10]. StAR is implemented as a Java library and acts as a wrapper to the service, enabling it to dynamically record assertions of provenance according to a policy file. StAR represents a benefit to the scientist in capturing assertions automatically.

Data is collected and *segmented*[4] to one of a set of possible values. This segmentation uses thresholds based on average values of the variables previously observed. The segmented value corresponds to an element within the pattern for a particular state as described in section 3. We use techniques from the Time Series Knowledge Representation (TSKR) [6] to determine the intervals over which the segmented series coincide with one another as shown in figure 2(b). Details of these series are then used as the content of an actor state p-assertion, along with a complete pattern description indicating all those conditions which hold over the series. Following recording of documentation, any actor may be query the provenance store.

A user can determine future states for a process based upon the states previously documented within a provenance store. For all states which are related to the same event, a transition table listing the probability of state transition given that observed event may be calculated. The most likely next state for an actor is the one with the highest probability value given to the current state. In cases where the two states (predicted and actual) do not match we use a similarity measure to find how similar those states are. It is a simple distance measure of each corresponding pattern value, shown in equation 1, where q and r are the two patterns being compared and p and t are the number of items in the patterns and the number of possible values for each of those items. The total number of possible states is p^t, though for any given process run not all states may be observed.

$$s = 1 - \frac{\sum_{n=0}^{p} |q_n - r_n|}{p \times (t-1)} \qquad (1)$$

The distance between two states therefore is the total measured error between each of the states pattern elements, divided by the maximum total distance possible for error on each of those elements. Our measure differs slightly from a distance measure such as the Levenshtein distance as such a measure is unable to distinguish between the degree of change in any one of our pattern elements, just that elements differ. As pattern elements correspond to numerically ordered ranges in our model, taking account of the difference in these elements is important. The overall similarity of a process against a comparison process is calculated from the product of the similarities of each state. This similarity value gives us a single measure of how similar the conditions under which each of the actors involved were operating were for two processes.

5 Evaluation

We now demonstrate two uses of documenting actor state for a process; 1) attempting to predict future actor properties for processes based on previously documented ones and 2) Reducing manual navigation of process documentation based on comparison of states over the monitored intervals. The workflow we use to demonstrate this was the subject of both the first and second provenance challenges [7]. It is used to create population-based brain atlases from high resolution anatomical data from the Functional Magnetic Resonance Imaging (fMRI) Data Center[2]. We use StAR to automatically record assertions of interaction and state to a provenance store for each of the services in the workflow. We focus on the last two services in the workflow, which convert an averaged brain image (determined from the average of intensities of MRI scans) gathered from a collection of high resolution anatomical data into graphics files showing slices of the brain. Actor state assertions identify the interval over which the actor is invoked (between request and response messages) based upon TSKR mined series from the segmented values using three variables: bytes in per second, one minute load average and the amount of buffered memory.

Each of the services in the workflow is hosted separately on a IBM JS20 blade machine (2 x 2.1GHz, 1.5GB RAM) and the provenance store for each of them is a Sun x2100 machine (1 x 2.2GHz, 4GB of RAM). When the workflow executes, a single action is performed by each of the services used and a set of states are recorded for it. We perform the process 1000 times, delaying subsequent invocations to allow the systems to recover. We do this as features observed in the variables may continue for longer than just the duration of a single action. For our state prediction evaluation, results are based on this experiment being performed twice.

Prediction of Future Actor Properties. A scientist is able to build a likely model for future properties using a transition table (as described in section 4) for each action. A state prediction is made for each actor after a single process is executed, based upon the last known state of that actor and the most common transition observed. Figure 3 shows the percentage of matches for predicted states to actual ones for our approach along with a history based and simple monte-carlo prediction. Each point represents the match rate for a single actor in the process. All machines consistently predicted states at a reasonably high success rate (50-85%), which was always above that of the monte-carlo predictions. The average trend indicates that as more transitions are observed, state becomes more difficult to predict in the future. This is due to the increased complexity of the model which is built when more transitions are found. It is likely that a more sophisticated analysis of the transition pattern leading to a state could further increase this success rate. Using this approach, the scientist executing the process is able to form a hypothesis detailing the most likely states to occur for each actor during future invocation of each action.

[2] http://www.fmridc.org/

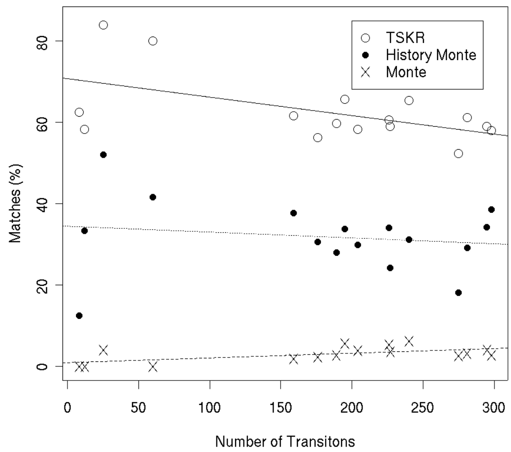


Fig. 3. Match rate of predicted states to those observed

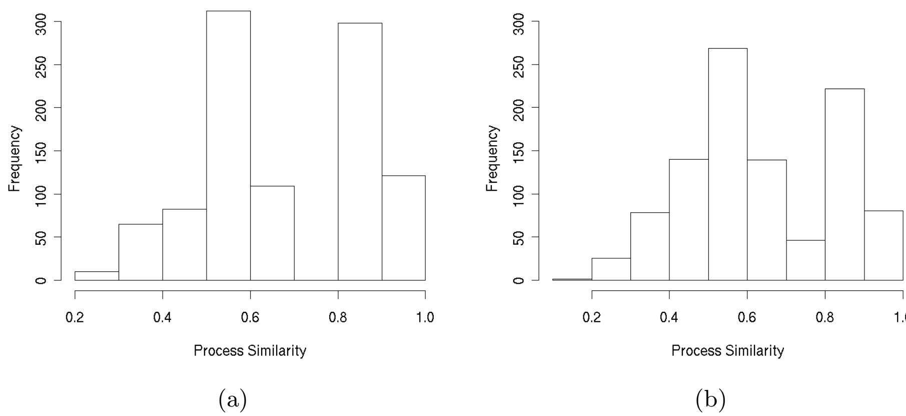


Fig. 4. Distribution of process similarity values when compared to model process

Reduction of Manual Process Documentation Navigation. We demonstrate reduction of navigation performed by a scientist by determining the similarity of a single state for each action within the process against that observed in a "comparison process", with our results shown in figure 4(a). The total similarity of a process (as defined in section 4) is the product of multiplying each of these similarities for each action. The scientist may then use these distances as a filter to locate the most interesting processes from a large collection of process documentation. Our results in figure 4 show the distribution of process similarity values. For our scenario, we are able to see that the total documentation to be navigated is reduced dramatically when searching for either those processes with a high or low similarity (≥ 0.9 or ≤ 0.4). This corresponds to 12% and 8% of all of the documentation recorded. If we look at the lowest similarity processes ($\leq$0.3), we can reduce this figure even further to 1% of all documentation. Figure 4(b) shows the same processes being compared, but with an average similarity value corresponding to the observation of multiple states within each invocation. We reveal a further number of interesting processes within the 0.1-0.2 range and 0.7-0.8 ranges by doing this, including even smaller subsets of documentation. Without the documentation of actor states, it is perfectly feasible that the navigation of records of all 1000 processes (totalling 56MB's worth of XML documentation to be queried in our own experiments) would have to be navigated manually.

6 Conclusion

In modeling actions performed by entities working as part of a process, strict event-based documentation may not be appropriate for documenting all process features. Instead, an interval based representation – such as the one presented in this paper, better represents observation of properties which hold true over a period of time. We have shown here that by documenting context of a process, it is possible to query provenance repositories to predict the future properties of actors or find other process traces which exhibit similarities to a model trace. Where vast collections of process documentation exist for the same workflow, being able to filter more interesting information for a scientist can present both time saving benefits and a reduction in the number of queries of documentation. In our evaluation we were able to reduce this to 1% of the overall captured documentation.

References

1. Brown, P.J.: The Stick-e Document: A Framework for Creating Context-aware Applications. Electronic Publishing - Origination, Dissemination, and Design 8(2/3), 259–272 (1995)
2. Dey, A.K., Abowd, G.D.: Towards a Better Understanding of Context and Context-Awareness. In: Workshop on The What, Who, Where, When, and How of Context-Awareness. ACM Press, New York (April 2000)
3. Groth, P., Miles, S., Moreau, L.: PReServ: Provenance Recording for Services. In: Proceedings of the UK OST e-Science second All Hands Meeting 2005, AHM 2005 (2005)
4. Keogh, E., Chu, S., Hart, D., Pazzani, M.: Segmenting Time Series: A Survey and Novel Approach (1993)
5. Miles, S., Groth, P., Branco, M., Moreau, L.: The requirements of using provenance in e-Science experiments (January 01, 2006)
6. Moerchen, F.: Algorithms for time series knowledge mining. In: KDD 2006: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 668–673. ACM, New York (2006)
7. Moreau, L., Ludäscher, B. (eds.): The First Provenance Challenge. Concurrency and Computation: Practice and Experience 20(5), 409–418 (2007)

8. Moreau, L., Freire, J., Futrelle, J., McGrath, R., Myers, J., Paulson, P.: The Open Provenance Model (December 2007), http://eprints.ecs.soton.ac.uk/14979/
9. Myers, J.D., Pancerella, C.M., Lansing, C.S., Schuchardt, K.L., Didier, B.T., Goble, C.N.A.: Multi-scale Science: Supporting Emerging Practice with Semantically Derived Provenance (March 06, 2006)
10. Wootten, I., Rajbhandari, S., Rana, O.: Automatic Assertion of Actor State in Service Oriented Architectures. In: ICWS 2007, pp. 655–662 (2007)
11. Zhao, J., Goble, C.A., Stevens, R.: An Identity Crisis in the Life Sciences. In: Moreau, L., Foster, I. (eds.) IPAW 2006. LNCS, vol. 4145, pp. 254–269. Springer, Heidelberg (2006)