# Towards the Explanation of Workflows

James R. Michaelis, Li Ding, and Deborah L. McGuinness

Tetherless World Constellation, Rensselaer Polytechnic Institute,
110 8th Street, Troy, NY 12180, USA
{michaj6, dingl, dlm}@cs.rpi.edu

**Abstract.** Across many fields involving complex computing, software systems are being augmented with workflow logging functionality. The log data can be effectively organized using declarative structured languages such as OWL; however, such declarative encodings alone are not enough to facilitate understandable workflow systems with high quality explanation. In this paper, we present our approach for visually explaining OWL-encoded workflow logs for complex systems, which includes the following steps: (i) identifying and normalizing provenance in workflow logs using the provenance interlingua PML2, (ii) using this provenance information, as well as supplemental log data, building an abstracted workflow representation known as a RITE network (capable of storing workflow state Relationships, Identifiers, Types, and Explanations), and (iii) visualizing the workflow log by displaying its provenance information as a directed acyclic graph and presenting supplemental explanations for individual workflow states and relationships. To demonstrate these techniques, we describe the design of a workflow explainer for the Generalized Integrated Learning Architecture (GILA) – a multi-agent platform designed to use multiple learners to solve problems such as resolving airspace allocation conflicts. We also comment on how our approach can be generalized to explain other complex workflow systems.

## 1 Introduction

Increasingly, application developers are making an effort to audit system activities using intensive logging, so as to provide increased insight into the underlying workflows of their systems. In addition to recording system activities, such workflow logs can explicitly capture both system operating contexts and capabilities. Inherently, this data can become very complex as system complexity increases, resulting in diminished intuitiveness for human auditors. One coping strategy is to facilitate the development of visualization-oriented interfaces aimed at helping auditors explore, understand and validate these complex workflow logs.

While most workflow logs are inherently well structured, this won't automatically make it easy for auditors to explore and interpret them. A small piece of structured domain data may not be hard to understand. However, a huge amount of such data could easily overwhelm a typical human's perception capacity. Moreover, the domain-specific structure of workflow data typically requires auditors to know a non-trivial amount of domain knowledge.

Complex workflow logs of this nature are generated by the Generalized Integrated Learning Architecture (GILA) [1] – a multi-agent platform designed to integrate results from multiple learning systems. It has been demonstrated on the task of resolving airspace allocation conflicts among aircrafts by learning from a small number of examples provided by a human domain expert. In GILA, each internal agent generates a large number of domain-specific log entries, encoded in the Web Ontology Language (OWL)[1], encompassing both collaborations with other agents and domain knowledge. This use of a best practice language allows for both effective information integration and structuring in the log data. Nonetheless, our preliminary analysis (conducted in coordination with the GILA development team) indicates its logs remain difficult to scan for things such as global agent interaction patterns.

Our intention is to find ways to help auditors explore complex workflow logs, encoded through domain-specific structures, and discover interesting patterns. We should mention that auditors may have expertise in the subject matter of a workflow system, but lack expertise in its software architecture and implementation. For example, GILA auditors might include air traffic controllers.

In this paper, we propose a general-purpose workflow explanation technique, which takes in a log and applies the following steps: (i) identifying a concise representation of provenance in the log , (ii) enriching the log data by extracting and normalizing provenance annotations using the Proof Markup Language v2.0 (PML2) [2], (iii) connecting the enriched log data with a general purposed data model for visually explaining workflows, known as a RITE network, and (iv) using the data from a RITE network to generate visual explanations for an auditor. To demonstrate this procedure, we have implemented a prototype system for explaining logs produced by GILA.

The remaining sections are organized as follows. Section 2 introduces the challenges underlying complex workflow logs as generated by GILA. Next, Section 3 details our approach to explain complex workflows, through our experiences with GILA. Following this, Section 4 surveys related work. Finally, Section 5 discusses our conclusions and intended future work. Where appropriate, we will discuss relevant highlights of our GILA workflow explanation prototype.

## 2 Motivating Challenges for Workflow Explanation in GILA

The Generalized Integrated Learning Architecture (GILA) is a general-purpose integrated multi-agent platform that solves domain problems by learning from a problem-solution pair submitted by a human expert. One of the key purposes of GILA is to learn how humans solve complex problems and apply this knowledge to future problems. A complex problem domain known as the Airspace Control Scenario has been chosen to drive the development of GILA and evaluate its performance. The objective of this problem domain is to resolve conflicts in a collection of airspace allocations for aircrafts.

[1] OWL: http://www.w3.org/TR/owl-features/

The example solutions provided by domain experts follow a certain pattern. First, the domain expert is given a collection of airspace allocations (3D spatial extent plus time range), each of which corresponding to an aircraft. With this information, she then tries to iteratively modify one airspace assignment at a time until no outstanding conflicts remain. In turn, GILA learns and mimics these observed problem-solving techniques. To accomplish this, a meta agent (known in GILA as the Meta-Reasoning Executive, or MRE) coordinates specialized learning agents (known as Integrated Learning Reasoners, or ILRs) to run the following routines:

1. List all outstanding conflicts within the problem state
2. Stop if no conflicts are found – otherwise, pick one conflict and ask the ILRs to suggest conflict resolution strategies (henceforth known as solutions)
3. Ask an alternate set of agents to evaluate each of the received solutions using different metrics. Some examples include, (a) the number of aircraft conflicts which would remain (this is important, since modifying one aircraft could indirectly resolve other conflicts it is part of), (b) potential safety constraint violations (pre-defined conditions which aircraft should avoid ending up in), and (c) cost of application (defined by calculations beyond the scope of this work).
4. Choose one solution based on internal metrics that take into account all the evaluation results. Each time a solution is chosen, a new problem state is produced to reflect the remainder of the problem.

The communications among GILA agents are independently recorded in OWL and aggregated into one central log on a communal blackboard. Ultimately, the GILA log itself is generated through combining these blackboard logs with a series of supporting OWL ontologies in the background knowledge.

### 2.1 Challenges Underlying the GILA Workflow

The GILA log serves as a good test case for our workflow explanation work, in that it contains many challenges present in other complex logs. Two of these are as follows:

1. **Non-intuitive workflow provenance.** The OWL ontologies for encoding the GILA log are primarily designed for the Airspace Control Scenario. This renders the workflow provenance too unintuitive for most people to explore and understand. In general, Non-intuitive provenance becomes increasingly problematic as user background on domain knowledge decreases.
2. **Data volume.** While each log entry (denoting one step of agent activity) is quite simple, step 3 in the GILA analysis routine (see above) introduces a huge amount of log entries with repetitive structuring. This is because, for each problem-solving iteration, each candidate solution for the problem needs to be evaluated by several metrics. On average, each run of the GILA system produces a log, encoded in OWL, on the order of 50,000 triples (or 22 megabytes) in size – large enough to cause problems for many Semantic Web data processing technologies. Though more important to us, the size of this dataset simply makes it harder for users to effectively browse. As such, the relevant workflow relationships usually end up getting obscured by a large amount of less important data for encoding complex domain information.

## 3 Our Workflow Explanation Process

This section discusses our domain-independent strategy for building workflow explainers, addressing the motivating challenges of GILA from Section 2.1.

- In order to address data volume issues, we adopt a strategy to seek out a concise provenance representation for a workflow log. Such a provenance representation may account for only a fraction of the log content. Nonetheless, it could provide a very intuitive representation of the overall pattern of a workflow. In turn, any additional information in the workflow log could be used to supplement explanations of this provenance later on.
- Different workflow systems will use different terminology and structuring for representing provenance relationships. Therefore, as a precursor to addressing issues of non-intuitive and non-interoperable provenance, we aim to normalize its encoding using the Proof Markup Language v2.0 (PML2).
- To help facilitate visualization of workflow data, we propose a data model backed by our normalized provenance known as a RITE network. RITE networks can supply a global picture of data usage in the execution of a workflow. In addition, they can help users retrieve detailed descriptions regarding the nature of individual workflow states. To create these workflow state descriptions, we support HTML/XSLT-based templates which are backed by SPARQL[2] querying over an enriched workflow log (see Section 3.3 for more details).
- To handle the rendering of RITE networks, we introduce a graph-based visualization strategy to facilitate their inspection by auditors. For this, directed acyclic graph (DAG) based formats seem appropriate for concisely expressing their contained provenance relationships.

The above methodology is designed to address complex workflows from various domains, such as multi-agent systems, composite scientific workflow systems, distributed policy reasoning systems, and hybrid logical reasoning systems. In any case, we assume that the target workflow log will be complete and encoded in OWL (otherwise, an ontology engineer can convert a structured log into an OWL representation). In what follows, we elaborate on the steps above and demonstrate how we apply each toward explaining the GILA workflow log.

### 3.1 Uncovering Provenance in Workflow Logs

According to our workflow explanation strategy, we seek to identify a concise provenance representation for a given log. We find two major advantages in doing this.

First, this helps us to focus on different aspects of the workflow. For instance, it may be prudent to only capture in a provenance representation the execution of workflow processes. Likewise, supplemental knowledge records could capture fine details about the data being handled by these processes.

---

[2] SPARQL: http://www.w3.org/TR/rdf-sparql-query/

Second, we see benefits from a user interface perspective. In theory, a provenance record encompassing an entire workflow log could be generated. However, depending on the workflow system in question, providing this much provenance information to an auditor could lead to usability problems.

To handle provenance identification in an OWL-based workflow log, the appropriate OWL classes and properties used to encode provenance information must be determined. This will allow for a mapping to a normalized provenance representation, as we discuss in Section 3.2. This step may require domain expertise and manual work – however, it is a one-time job for each workflow log. In addition, once this is done, instances of relevant classes in the original workflow log could implicitly become instances of the normalized provenance classes via OWL inference.

In the GILA system, an ontology is defined for capturing provenance information through the OWL-based GILA inter-component language (GIL), known as *gilcore* (the core GILA ontology). Figure 1 shows both the class tree and the property tree for *gilcore*, as viewed in the SWOOP[3] ontology browser. Taking some examples from below, the class *gilcore:Solution* records an agent's activity in generating a solution to a problem, and the property *gilcore:hasProblem* denotes a dependency between the solution generation activity and problem generation activity of alternate agents. In the following section, we discuss our procedure for mapping this information to PML2.

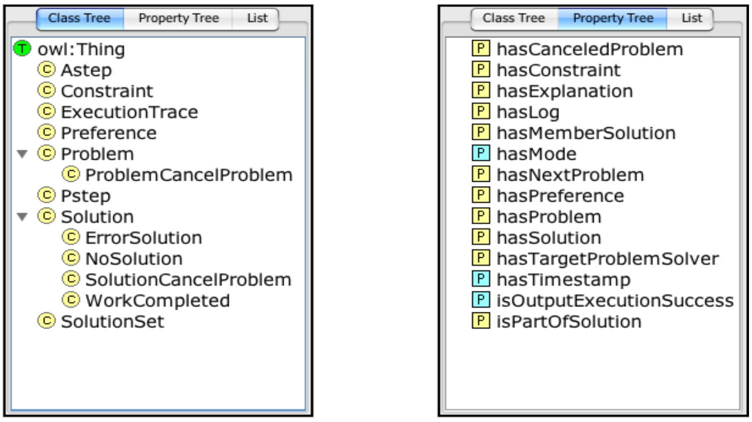



**Fig. 1.** Class tree (Left) and property tree (Right) for gilcore ontology

### 3.2 Normalizing Workflow Provenance using PML2

Since a workflow log in its raw form may contain only provenance encoded by domain specific classes and properties, it may be difficult for auditors to understand. Therefore, once a concise provenance representation is identified, we normalize it to make it easier to process and understand. We focus on normalizing provenance information by annotating the original log data with generic provenance encoded in PML2. Figure 2 gives an overview of the relationships present in PML2.

3 SWOOP: http://www.mindswap.org/2004/SWOOP/

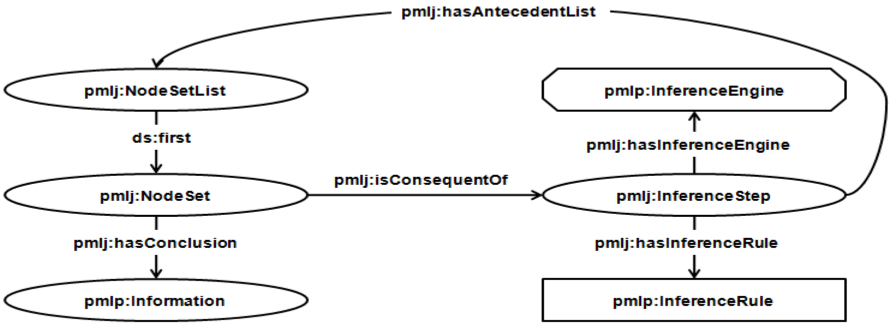



**Fig. 2.** Relationships defined in PML2

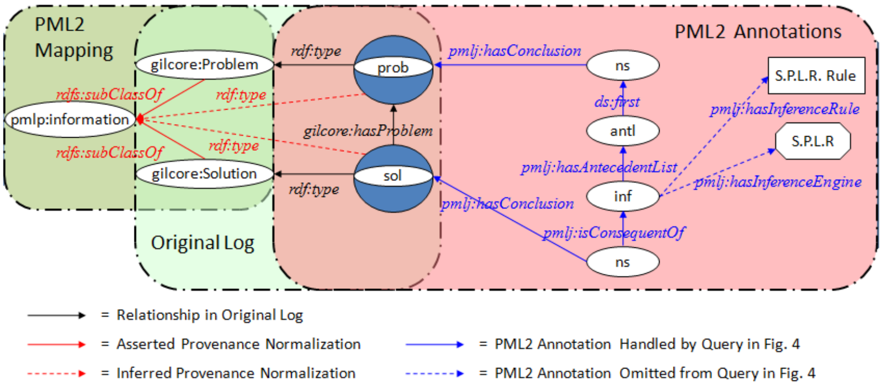



**Fig. 3.** An example PML2 extension for the original GILA log

The normalization process is enabled by adding PML2-based information to the original log using OWL inference (e.g. subclass reasoning) and SPARQL based rules. Figure 3 shows an example of a PML2 encoding which extends a section of the GILA log, and can be read as: Agent "S.P.L.R." coordinated event "inf" with operation "S.P.L.R. Rule" using the input data "prob" and outputting the data "sol". This encoding works as follows:

- First, the OWL classes *gilcore:Solution* and *gilcore:Problem* are asserted to be subclasses of the OWL class *pmlp:Information*. Through OWL inference, instances of these two classes will be inferred to be of type *pmlp:Information* .
- Next, an instance of *gilcore:Solution* (sol) is identified. This instance has an asserted property *gilcore:hasProblem,* which points to a node of type *gilcore:Problem* (prob). This is known based on the property's range restrictions.
- Following this, a corresponding PML encoding is then created to normalize the (sol, *gilcore:hasProblem*, prob) provenance relationship. We accomplish this through a SPARQL query like the one in Figure 4. For purpose of conciseness, we omit from the query here any information on agent S.P.L.R. itself.

```
PREFIX rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX pmlj:    <http://inference-web.org/2.0/pml-justification.owl#>
PREFIX ds:      <http://inference-web.org/2.0/ds.owl#>
PREFIX gilcore: <http://inference-web.org/2006/12/gilcore.owl#>
CONSTRUCT      { _:x a pmlj:NodeSet .
                 _:x pmlj:hasConclusion ?solution .
                 _:x pmlj:isConsequentOf _:y .
                 _:y pmlj:hasAntecedentList _:z .
                 _:z ds:first _:w .
                 _:w pmlj:hasConclusion ?problem }
FROM           <http://inference-web.org/data/gilalog.owl>
WHERE          { ?solution a gilcore:Solution .
                 ?solution gilcore:hasProblem ?problem }
```

**Fig. 4.** SPARQL query for generating PML2 annotations in Figure 3

### 3.3 Connecting the Workflow Log to Explanation Routines with RITE Networks

With our routines for normalizing provenance in a workflow log established, we now need a data model for expressing workflow information for later presentation. To give a better idea of what this model requires, it is important for us to state what we feel should be presented to an auditor. We currently seek to handle workflow log explanations in two parts: a provenance-based workflow graph (the provenance component), and supplemental data explanations for individual states in that graph (likewise, the supplemental component).

To facilitate this, we introduce in this paper a new data model called a RITE network. Each instance of a RITE network can be defined as a graph $G = (S, R)$, with a set of states $S$ having the following properties:

- *T:* A defined type, which corresponds to the original RDF type of a corresponding entity in the workflow log (e.g., *gilcore:solution*).
- *I:* A unique identifier, which corresponds to a URI defined in the provenance normalized workflow log.

To link these vertices together, a set of directed binary relationships $R$ are defined. In essence, these relationships can only be formed between exactly two vertices (a source, and target) and can only apply in one direction.

Finally, for each node in $S$ and relationship in $R$, an accompanying pair of explanation templates $E$ is defined: a label template, and a supplemental explanation template. The label template defines a label to be written directly on a node or relationship in the visualized provenance graph. This simply consists of a section of HTML/XSLT code for displaying: (i) for nodes, the type and/or identifier, or (ii) for relationships, their name. Likewise, the supplemental template defines a secondary (more detailed) explanation for nodes and relationships not captured by the provenance graph. This template contains two things: a SPARQL query for retrieving relevant data from the workflow log (since it must be OWL encoded), and a section of HTML/XSLT code for formatting the retrieved data into a user friendly format. More details on this will be given in Section 3.4.

The RITE network relationships mentioned here are presented in Figure 5.

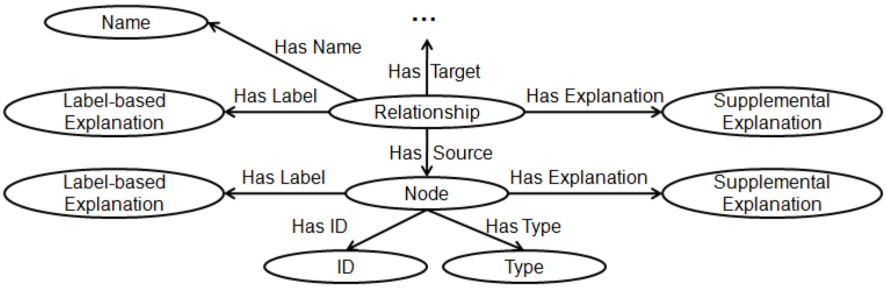



**Fig. 5.** Concept map for RITE network

### 3.4 User Interface Issues: Presenting the RITE Networks

Since our main goal is to help auditors inspect workflow logs, we will now discuss the user interface design techniques required to display the information in RITE networks. These approaches address the limitations presented by both computer monitors and human perception capacity.

First, we discuss our strategy for rendering the general graph structure of a RITE network. Currently, many provenance explanation systems rely upon tree-based rendering. However, we instead use a directed acyclic graph (DAG) format. In reaching this decision, two key properties of the GILA workflow log were considered: (i) in many locations, multiple work states may share a common (direct or indirect) antecedent or a common (direct or indirect) consequent, and (ii) no circular provenance relations (e.g., workflow state A depends on workflow state B, and vice versa) could be found. For the GILA log, after narrowing down the original 50,000 triples, we were able to generate a 150 node DAG to be visualized in the final rendering. Since trees do not allow multiple nodes to share a single antecedent, we predicted an exponential increase in required nodes for each added problem solving iteration in the workflow.

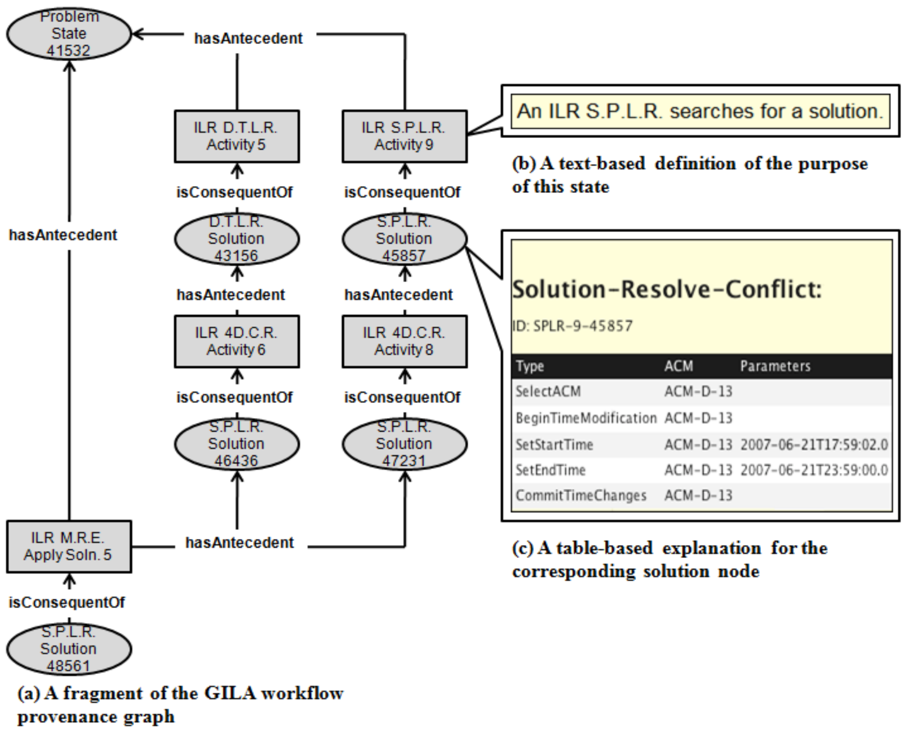
Now that we have defined how to render a RITE network's graph structure, we will discuss our strategy for rendering its individual nodes and relationships. For each of these, we write a brief caption as defined by its label template. As mentioned previously, this label is intended to display: (i) for a node, its type and/or identifier, or (ii) for a relationship, its name. Following this, we assign different cell shapes to differentiate types of RITE nodes. Currently, we use ovals to represent inferred instances of *pmlp:Information*, and rectangles for inferred *pmlj:InferenceStep* instances (for details on PML2 type inference, see section 3.2). Second, we organize the nodes on a grid layout, based on their provenance relations. Nodes occurring before a given node will go on a grid row above, nodes occurring after will go below, and nodes occurring at the same time will stay on the same layer. We present an example of this rendering style in Figure 6(a), which represents a small section of a GILA workflow. Finally, for each edge, we simply print the name of the appropriate property on (or near) it.

With this, we have addressed both key presentation issues in the provenance component of the RITE network. Now, we proceed to address the supplemental component – determining intuitive explanations for individual nodes and relationships. As part of this, we will discuss some more of the rationale for incorporating supplemental explanations into RITE networks.

In general, a provenance graph layout itself can help users understand general workflow patterns. However, it typically isn't very good for highlighting the properties of individual components. Indeed, it appears quite possible that even someone with a fair amount of experience with GILA could become confused by the presence of certain nodes or relationships in one of its provenance graphs.

Two varieties of node-based supplemental explanations are given in Figures 6(b) and 6(c). In Figure 6(b), a simple text-based explanation is presented for its corresponding node. Likewise, Figure 6(c) details the content of an S.P.L.R. solution (also known in GILA terminology as a Solution-Resolve-Conflict) in terms of a table-based presentation of the proposed modifications to aircraft ACM-D-13.

Naturally, the formatting of information within the node explanations should vary based on what needs to be rendered. As can be seen in Figure 6(c), a table was necessary to render the properties of each step in the solution. Likewise, no such complex formatting was necessary to present the simple definition given in Figure 6(b). Finally, to handle user access to supplemental explanations via the general workflow visualization, we recommend this be addressed on a case-by-case basis (for our GILA explanation prototype, we provide access to these explanations via tooltips).




**(a) A fragment of the GILA workflow provenance graph**

**(b) A text-based definition of the purpose of this state**

**(c) A table-based explanation for the corresponding solution node**

**Fig. 6.** Example rendering of RITE network for GILA workflow log fragment

Figure 7 illustrates how we leverage SPARQL and XSLT to build an explanation template for the explanation in Figure 6(b). In this case, a SPARQL query is executed to extract information from the normalized workflow log, and an XSLT template is used to generate HTML from the XML encoded SPARQL result[4].

Admittedly, no single type of explanation template for a given node in the GILA workflow will be ideal for every user. As such, an eventual goal of ours will be to allow users to define custom explanation templates. More details on this are given in the Conclusions and Future Work section.

**QUERYING:**

```
PREFIX rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX pmlj:    <http://inference-web.org/2.0/pml-justification.owl#>
PREFIX pmlp:    <http://inference-web.org/2.0/pml-provenance.owl#>
PREFIX gilalog: <http://inference-web.org/data/gilalog.owl#>
SELECT ?name
WHERE  {
         ?nodeset pmlj:hasConclusion (ILR S.P.L.R. Activity 9) .
         ?nodeset pmlj:isConsequentOf ?infStep .
         ?infStep pmlj:hasInferenceEngine ?engine .
         ?engine pmlp:hasDescription ?description .
         ?description pmlp:hasRawString ?name . }
```

**RENDERING (XSLT Template):**

```
… An ILR <xsl:value-of select="//@name"/> searches for a solution. …
```

**Fig. 7.** An example supplemental explanation template for Figure 6(b)

## 4 Related Work

We have identified two key works which leverage provenance to explain RDF/OWL based workflow information.

The first of which, Provenance Explorer [3], is designed to take in OWL-based workflow data, and generate a corresponding graph of the contained provenance information. We see our workflow explanation technique as having two advantages. First off, it appears that only our work attempts to provide a normalized representation of workflow provenance (in the form of PML2). We saw this as being an important step to address issues involving non-intuitive provenance (see Section 2.1). Additionally, our supplemental workflow state explanations (versus drill-down provenance representations in Provenance Explorer) can be a potential solution to the scalability issues auditors may face in working with highly complex workflow data.

The second work, which serves as a direct precursor to our efforts, is the Inference Web Framework [4]. This framework is designed to provide a domain-agnostic provenance model (PML2), as well as tools for exploring and validating PML2 data. While the Inference Web PML2 browser, IWBrowser, is able to visually present small datasets, problems emerge with data scalability. Again, we see our usage of supplemental workflow explanations as being a key to addressing such issues.

4 SPARQL Query Results XML Format: http://www.w3.org/TR/rdf-sparql-XMLres/

Moving on, an alternate area somewhat relevant to our work deals with the design of tools for graphically manipulating provenance in complex workflows. Two examples of such systems are Taverna [5], which is designed to represent in silico bioinformatics experiments, and Triana [6], likewise designed to track web service compositions. We see our work as different from these approaches in that we are not focused on having users build their own workflows. Rather, we want to present workflows derived from complex RDF data for explanation to end users.

One final area of work related to ours deals with frameworks for capturing provenance metadata from workflows [7] [8]. However, like with Taverna and Triana, these works seem to focus on tracking provenance through user-defined workflow data.

## 5 Conclusions and Future Work

In this paper, we presented a set of techniques for both extracting data from workflow logs, and presenting it to an auditor in an understandable form. We view this methodology as being capable of processing and explaining complex workflow data in various domains, such as multi-agent systems, composite scientific workflow systems, distributed policy reasoning systems, and hybrid logical reasoning systems. Following our design of an explanation interface for GILA, we are using two additional complex workflow-generating systems to vet our approach. These include the Transparent Accountable Data-Mining Initiative (TAMI) [9] and Semantic Provenance Capture in Data Ingest Systems (SPCDIS) [10]. The transaction logs in TAMI and SPCDIS seem to provide promising corpora for testing our methodology and infrastructure.

Our combined usage of concise workflow provenance and supplemental explanations for workflow states both show initial promise. However, different users may require nodes be explained differently. Therefore, one of our goals in future work will be twofold. First, we hope to provide auditors with the ability to craft their own concise provenance representations without the support of an outside specialist. Similarly we hope to allow auditors to write the querying and formatting components of their own explanation templates (or at least give them more flexibility in how the templates are configured).

Another issue deals with the formulation of initial provenance graphs in the RITE networks. While we were able to represent the 50,000 triples in the GILA log through a 150 node DAG, presenting this visualization as a whole was not without challenges. For instance, with the current version of the GILA explanation program, a user can navigate the workflow either through vertical scrolling or basic zooming functionality we provided. The latter of these seems like an attractive solution to get the entire workflow in one screen. However, when this is done, nodes and edges become far too small to effectively label via our rendering approach. To address this issue, we are considering an extension to our RITE data model to allow for a combination of hierarchical grouping of nodes with supplemental explanations. Through this, zooming thresholds could be set such that different levels of a RITE network hierarchy could be viewed at different magnifications.

Finally, we are currently looking into ways to conduct usability studies on the workflow explanations generated by our methodology. We hope to generate workflow explanation tools for several complex systems. In turn, we would like to evaluate user experience with these systems, based on varying degrees of user knowledge of the target domain. However, more concrete metrics on which to base these studies have yet to be established.

## Acknowledgments

This work is partially supported by NSF Award #0524481, IARPA Award #FA8750-07-2-0031, DARPA award #FA8750-07-D-0185, #55-002001, #FA8650-06-C-7605.

## References

1. Zhang, X. et al.: An Ensemble Learning and Problem Solving Architecture for Airspace Management. Accepted for publication by IAAI'09 (2009)
2. McGuinness, D. L., Ding, L., da Silva, P., Chang, C.: PML 2: A Modular Explanation Interlingua. In Proceedings of the AAAI'07 Workshop on Explanation-Aware Computing (2007)
3. Cheung, K. and Hunter, J.: Provenance Explorer – Customized Provenance Views Using Semantic Inferencing. In Proceedings of Fifth International Semantic Web Conference (ISWC), p. 215-227 (2006)
4. McGuinness, D. L. and da Silva, P.: Explaining answers from the Semantic Web: The Inference Web Approach. Journal of Web Semantics. Vol. 1 No. 4., p. 397-413 (2004)
5. Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M. R., Wipat, A., Li, P.: Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics. Vol. 20 No. 17., p. 3045-3054 (2004)
6. Majithia, S., Shields, M. S., Taylor, I. J., Wang, I.: Triana: A Graphical Web Service Composition and Execution Toolkit. In Proceedings of the IEEE International Conference on Web Services (ICWS'04), 514-524 (2004)
7. Bowers, S., McPhillips, T., Ludäscher, B., Cohen, S., Davidson, S. B.: A Model for User-Oriented Data Provenance in Pipelined Scientific Workflows. In Proceedings of International Provenance and Annotation Workshop (IPAW'06), Chicago, Illinois, USA (2006)
8. Simmhan, Y. L., Plale, B., Gannon, D. 2006. A Framework for Collecting Provenance in Data-Centric Scientific Workflows. In Proceedings of the IEEE International Conference on Web Services (ICWS) (2006)
9. Weitzner, D.J., Abelson, H., Berners-Lee, T., Hanson, C.P., Hendler, J., Kagal, L., McGuinness, D. L., Sussman, G. J., Waterman, K. K.: Transparent Accountable Inferencing for Privacy Risk Management. Proceedings of AAAI Spring Symposium on The Semantic Web meets eGovernment. AAAI Press, Stanford University, USA (2006)
10. McGuinness, D. L., Fox, P., da Silva, P., Zednik, S., Del Rio, N., Ding, L., West, P., Chang, C.: Annotating and embedding provenance in science data repositories to enable next generation science applications. American Geophysical Union, Fall Meeting 2008, abstract #IN11C-1052 (2008)