Inference Web in Action: Lightweight Use of the Proof Markup Language

Paulo Pinheiro da Silva[1], Deborah McGuinness[2]
Nicholas Del Rio[1], and Li Ding[2]

[1] University of Texas at El Paso, El Paso TX 79902, USA
[2] Rensselaer Polytechnic Institute, Troy NY 12180, USA

Abstract. The Inference Web infrastructure for web explanations together with its underlying Proof Markup Language (PML) for encoding justification and provenance information has been used in multiple projects varying from explaining the behavior of cognitive agents to explaining how knowledge is extracted from multiple sources of information in natural language. The PML specification has increased significantly since its inception in 2002 in order to accommodate a rich set of requirements derived from multiple projects, including the ones mentioned above. In this paper, we have a very different goal than the other PML documents: to demonstrate that PML may be effectively used by simple systems (as well as complex systems) and to describe lightweight use of language and its associated Inference Web tools. We show how an exemplar scientific application can use lightweight PML descriptions within the context of an NSF-funded cyberinfrastructure project. The scientific application is used throughout the paper as a use case for the lightweight use of PML and the Inference Web and is meant to be an operational prototype for a class of cyberinfrastructure applications.

1 Introduction

In a question-answering context such as when querying an intelligent agent, answer justifications are used to provide provenance information about the sources and process used by agent (or agents) producing the answers. The Proof Markup Language (PML) is a powerful language for encoding OWL-based justifications that are portable and combinable and that can be distributed over the web [12]. PML also facilitates agents in reusing elements of one justification as part of new justifications, enabling in this way multiple justifications for a single answer to be encoded within a single web artifact. Furthermore, PML design is grounded on proof theory, which enables it to encode formal proofs as justifications. As a consequence of these and many other advanced features of PML, many potential users of PML have not further considered the use of the language and its supporting Inference Web Infrastructure due to perceived complexity issues.

Despite the richness of constructs supporting some of the advanced features mentioned above, PML does not require justifications to be distributed, to be combined, or to be formal proofs. In this paper, we introduce a use case that

A. Sheth et al. (Eds.): ISWC 2008, LNCS 5318, pp. 847–860, 2008.

describes how potential PML users can benefit from a simpler, restricted set of PML constructs to encode very basic justifications as well as to a restricted set of Inference Web tools to perform useful tasks such as the retrieval and browsing of provenance information. We claim that this use case is representative of a set of needs that a broad range of applications face, and further that a broad range of users may similarly use a subset of PML and the Inference Web toolkit to address common problems related to explanation and trust recommendations.

Any simplification of an answer justification may have consequences in terms of missing information that may prove to be critical to support some justification-based tasks such as trust and uncertainty management or even just to verify the justification correctness. For example, in the process of an inference engine encoding a formal proof for a theorem, the omission of discharged assumptions in a single step of the proof may be reason enough for one to consider the entire proof to be unsound. However, such encoding is still a justification capable of identifying the set of axioms used to derive the theorem as well as the collection of information sources asserting the axioms. Thus, it is clear that this unsound proof is still better than no justification at all. With this notion of justification usefulness in mind, the paper describes how lightweight uses of PML have successfully been used to support the inspection of gravity maps as part of the NSF-funded Cyber-ShARE Center of Excellence on Cyber-Infrastructure[1].

The rest of this paper is organized as follows. Section 2 introduces a scientific application as a use case for lightweight use of PML. Section 3 revisits many aspects of PML including its relationship to the Inference Web infrastructure. Section 4 describes some strategies to simplify the process of instrumenting an application to generate PML. This section also highlights how some tools can be easily used in combination with PML. Based on the lightweight use of PML, Section 5 described the results of a user study that identifies the scientists' need of provenance to understand the results of the scientific application introduced in Section 2. Section 6 discusses further strategies to simplify the use of PML as well as describing related work. Section 7 summarizes the main contributions of the paper.

2 A Use Case for Provenance

In this section, we introduce an exemplar scientific use case that has a number of common provenance requirements. Scientists may use gravity data to get a rough idea of the subterranean features that exist within some region. Geoscientists are often only concerned with anomalies, or spikes in the data, which often indicate the presence of a water table or oil reserve. However these anomalies have the potential to be artificial and simply imperfections introduced during the data retrieval process including for instance some data merging and filtering techniques. With the use of provenance, however, one may be able to inspect the process used to retrieve data and this figure out potential sources of imperfections.

[1] http://cybershare.utep.edu

This process, which begins by scientists providing a region or interest or footprint, specified in terms of latitude and longitude, is defined by the sequence of tasks below:

1. gather raw point data (possibly from multiple sources) in the region
2. merge point data coming from distinct sources
3. filter raw point data to remove duplicate data
4. create a uniformly distributed dataset by applying a gridding algorithm
5. create a contoured rendering of the uniformly distributed dataset

In a cyber-infrastructure setting, each one of the five tasks above can be realized by a web service. This set of web services is piped or chained together; the output of one service would be forwarded as the input to the next service specified in the *workflow*, such as in [7].

In these types of situations where multiple workflows can satisfy a single type of request, the set of results generated by each workflow are returned to the scientist. As in any question/answer scenario, it is up to the scientist to determine what result to use. However, this situation is no different from how users interact with Web search engines. A single query often yields thousands of results, yet the burden is placed on the user to determine which answer is most appropriate. This is one of the main reasons that applications should be able to explain their results as further discussed in the following section.

3 Inference Web and the Proof Markup Language

The Inference Web [9,10] is a knowledge provenance infrastructure for conclusions derived from inference engines which supports interoperable explanations of sources (i.e. sources published on the Web), assumptions, learned information, and answers associated with derived conclusions, that can provide users with a level of trust regarding those conclusions. The ultimate goal of the Inference Web is the same as the goal of the gravity data scenario which is to provide users with an understanding of how results are derived by providing them with an accurate account of the derivation process and the information sources used (i.e. knowledge provenance [13]).

Inference Web provides PML to encode justification information about basically any kind of response produced by agents. PML is an RDF based language defined by a rich ontology of provenance and justification concepts which describe the various elements of automatically generated proofs. Without getting into the details of the main concepts supporting PML because of obvious reasons, we can say that PML justifications are graphs with the edges always pointing towards the final justification conclusion. PML justifications can also be used to store provenance about associated information sources. PML itself is defined in OWL [1] thus supporting the distribution of proofs throughout the Web. Each PML component, which is not yet defined here, can reside in a uniquely identified document published on the Web separately from the others.

4 Lightweight Use of the Inference Web

Any conclusion may have no justification, one justification, or multiple justifications. In the case of a scientist using a gravity map, it is clear that the map was generated from data provided by some sources, e.g., data points or annotations about the region of interest, and by some process, whether the process is computer-based or not. In this case, the map is the conclusion of a justification and the justification is a description of the process used to derive the map. The scientist's knowledge about the map, however, may be restricted to the fact that the map came from Book A. In this case, the scientist can still state that the map is asserted by Book A (or even by the authors of Book A). Both justifications for the map are legitimate and can be encoded in PML. Moreover, PML has been designed to encode all sorts of justifications including the combination of alternate justification for a given conclusion. Because of that, PML has a rich but rather complex set of constructors to encode justifications.

One of the goals of this paper is to demonstrate a lightweight use of PML that relies on three *simplification assumptions* listed below. Please note that lightweight use of PML does not preclude a later enhancement of PML documents that may benefit from the full provenance encoding power of PML.

Simplification Assumption 1 - No use of alternate justifications. The encoding of a single justification for a given conclusion implies that the justification can be considered as a single DAG of **nodes** connected by `hasAntecedent` relationships. In this situation, lightweight use of PML is achieved by considering each node in the DAG to be a single **PML node set** with a single **PML inference step**. The hasAntecedent relationship of the node is the hasAntecedent property of the only inference step inside each node set.

```
<rdf:RDF>
  <NodeSet rdf:about="http://iw.cs.utep.edu//contourMapPS_7355.owl#map">
    <hasConclusion>
      <pmlp:Information>
        <pmlp:hasFormat rdf:resource="http://iw.cs.utep.edu/registry/FMT/ps3.owl#ps3"/>
        <pmlp:hasRawString rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
          (*** THE BASE64 ENCODING OF THE POSTSCRIPT OF THE MAP GOES HERE ***)
        </pmlp:hasRawString>
      </pmlp:Information>
    </hasConclusion>
    <isConsequentOf>
      <InferenceStep>
        <hasIndex rdf:datatype="http://www.w3.org/2001/XMLSchema#int">0</hasIndex>
        <hasAntecedentList>
          <NodeSetList>
            <ds:first rdf:resource="http://iw.cs.utep.edu/griddedData7035.owl#gridmap"/>
          </NodeSetList>
        </hasAntecedentList>
      </InferenceStep>
    </isConsequentOf>
  </NodeSet>
</rdf:RDF>
```

In the example above , we show the last step of our gravity map workflow (step 5 of the use case). The final conclusion of the gravity map workflow is a

contour map identified by the URIRef ending on `#map`. In the node, the map itself goes in the `hasRawString` property of the node set. The inference step of the node has that it has been derived from the conclusion of the node identified by the URIRef ending on `#gridmap` (the `#gridmap` URIRef corresponds to step 4 of the use case). The *hasFormat* attribute of a node set is optional as are many other node set and inference step properties. In the case of the contour map node, the `hasFormat` attribute says that the raw string is encoded in PostScript 3. URIs in the fragments of PML documents used in this section have been modified to fit in the paper. Complete PML documents in support of the gravity map use case are available at http://iw.cs.utep.edu:8080/service-output/proofs/.

Simplification Assumption 2 - No knowledge about the inference mechanism used to transform information in a step of a given information manipulation process. In a formal proof, it is expected that one can identify the inference rule used, e.g., resolution, or algorithm, e.g., quick sort, used in each step of the proof. PML can indeed be used for encoding formal proofs but it is often used to encode less structured justifications often called *information manipulation traces*. In concrete terms, it is common that the person instrumenting a process to generate PML may not have full knowledge about the process so that this person can properly document how information is transformed along the process. In the example below, we show how informal but still useful metadata can be added to the process trace of the gravity map. In this case, it is known that the service is a generic service, as identified by the inference rule identified by the URIRef ending on `#genericService` and that the service is named "contour" (as identified by the inference engine identified by the URIRef ending on `#contour`). These URIRef have been created by the example in this paper and they can be reused or new ones can be created on demand. These URIRef are called provenance elements and are also PML documents. In the past, Inference Web used to register these provenance elements in order to facilitate reuse. Currently, Inference Web still incentive the reuse of these documents but also allows these elements to be easily created and stored locally. The IWSearch capability described in Section 4.2 is used instead of the registry to facilitate the location and reuse of PML documents.

```
<rdf:RDF>
   (...)
    <isConsequentOf>
      <InferenceStep>
        <hasIndex rdf:datatype="http://www.w3.org/2001/XMLSchema#int">0</hasIndex>
        <hasInferenceRule
             rdf:resource="http://iw.cs.utep.edu/registry/RUL/GS.owl#genericService"/>
        <hasInferenceEngine
             rdf:resource="http://iw.cs.utep.edu/registry/IE/contour.owl#contour"/>
        <hasAntecedentList>
          <NodeSetList>
            <ds:first rdf:resource="http://iw.cs.utep.edu/griddedData7035.owl#gridded"/>
          </NodeSetList>
        </hasAntecedentList>
      </InferenceStep>
    </isConsequentOf>
   (...)
</rdf:RDF>
```

Simplification Assumption 3 - No knowledge about how information has been asserted from a given source. Conclusions of the leaf nodes in a justification DAG are pieces of information that have been asserted by some source. For example, in the case of the gravity map, the information may correspond to the gravity reading data points that was eventually processed, e.g., gridded, to generate the contour map, where the entire gravity database is the source. In reality, a web service was used to access the database over the web and some additional parameter where required to invoke the service. Leave nodes are called *direct assertions* since they make use of the direct assertion inference rule, i.e., the PML instance identified by the URIRef ending on `#told`. The example below illustrates the use of a **PML SourceUsage** instance often attached to direct assertions. Source usage is a complex concept since it has a rich set of properties used to specify how exactly a given piece of information was extracted from a given source. In the case of a lightweight use of source usage, however, we restrict its use to the identification of the source that was used without identifying how and when it was used. For instance, the source usage in the example below identifies that the gravity database identified by the URIRef ending on `#database` was used as a source of the conclusion of the node. It is interesting to note that PML identifies the agent responsible for retrieving the source that is identified by the value of the `hasInferenceEngine`.

```
<rdf:RDF>
   (...)
    <isConsequentOf>
      <InferenceStep>
        <hasIndex rdf:datatype="http://www.w3.org/2001/XMLSchema#int">0</hasIndex>
        <hasInferenceRule rdf:resource="http://iw.cs.utep.edu/registry/RUL/Told.owl#told"/>
          <hasSourceUsage>
            <pmlp:SourceUsage>
               <pmlp:hasSource
                  rdf:resource="http://iw.cs.utep.edu/registry/PER/GravityDB.owl#database"/>
            </pmlp:SourceUsage>
          </hasSourceUsage>
        <hasInferenceEngine
           rdf:resource="http://iw.cs.utep.edu/registry/IE/AccessDatabase.owl#accessDB"/>
      </InferenceStep>
    </isConsequentOf>
   (...)
</rdf:RDF>
```

4.1 PML Service Wrapper (PSW)

The encoding of lightweight PML is such a straightforward task that we have created a wrapper that can automate the generation of PML justifications for web services [4]. In the case of workflows based on web services, the use of the wrapper may allow PML to be used on the same way provenance is used in workflow-centered infrastructures, as later discussed in Section 6.

The gravity map scenario is realized by a service-oriented workflow composed of five Simple Object Access Protocol (SOAP) services, which gather, merge, filter, grid and render gravity datasets respectively[2]. These Web services are piped

[2] The gravity map workflow is available for use at http://iw.cs.utep.edu:8080/service-output/probeit/clientapplet.html

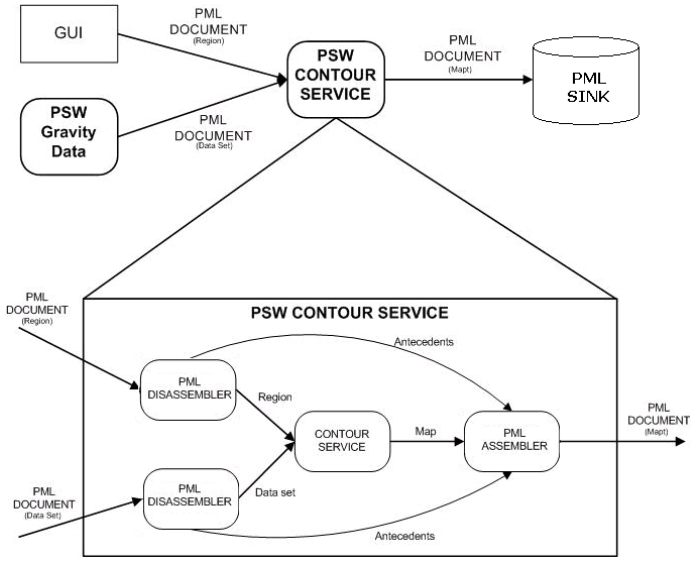

Fig. 1. Example of PSW configured for a contouring service

or chained together; the output of one service is forwarded as the input to the next service specified in the workflow. A workflow director is responsible for managing the inputs/outputs of each service as well as coordinating their execution. Provenance associated with scientific workflows of this nature might include the service's execution sequence as well as each of their respective outputs, which we refer to as *intermediate results*.

PML Service Wrapper (PSW) is a general-purpose Web service wrapper that logs knowledge provenance associated with workflow execution as PML documents. In order to capture knowledge provenance associated with workflows execution, each service composing the workflow has an associated PSW wrapper that is configured to accept and generate PML documents specific to it. Since PML node sets include the *conclusion* element, which is used to store the result of an inference step or Web service, the provenance returned by the wrappers also includes the service output thus workflows can be composed only of these PSWs; this configuration introduces a level of indirection between service consumers (i.e. workflow engine) and the target services that performs the required function. In this sense, PSW can be seen as a server side provenance logger.

The logging capability provided by PSW can be decomposed into three basic tasks: decompose, consume, and compose as illustrated in Figure 1. Upon invocation, the wrapper decomposes the conclusion of an incoming PML document, i.e., extracts the data resident in the PML conclusion using Inference Web's PML API. PSW then consumes the target service, forwarding the extracted data as an input to the target service. The result and associated provenance of the target service is then composed to produce the resultant PML document,

the PSW output. For example, a contouring service requires 2D spatial data to map and the region to consider in the mapping therefore a PSW wrapper for this contouring service would require two PML documents, one containing 2D spatial data, coming from some data retrieval service, and the other containing a region, (e.g. specified by latitude and longitude) specified by some user. The output of the contour service is a map, from which a new PML document is created, referencing the two input PML node sets as antecedents.

PSW has been developed in support of scientific workflows able to execute in a distributed environment such as the cyberinfrastructure. In traditional Inference Web applications [11,10], inference engines are instrumented to generate PML. However in a cyberinfrastructure setting, reasoning is not necessarily deductive and is often supported by Web services that can be considered "black boxes" hard to be instrumented at source-code level to generate PML. This is the primary reason why PSW, a sort of external logger, must be deployed to intercept transactions and record events generated by services instead of modifying the services themselves to support logging. Despite this apparent limitation, PSW is still able to record provenance associated with various target systems' important functions. For example, PSW configured for database systems and service oriented workflows can easily record provenance associated with queries and Web service invocations respectively in order to provide a thorough recording of the provenance associated with cyberinfrastructure applications.

4.2 Inference Web Search

IWSearch is developed to facilitate users accessing PML data distributed on the Web. In the case of our use case, the PML data correspond to the values of PML properties such as `hasInferenceRule`, `hasInferenceEngine` and `hasFormat`, as presented in Section 4.

In the past deployments of Inference Web, the provenance metadata are stored in a federated online repository IWBase, which provides both a web user interface and a SOAP web service interface for Inference web users to publish provenance metadata. IWSearch is motivated by the limitations of such provenance data management found in our past practice: (i) IWBase provides limited mechanisms for accessing registered metadata entries, i.e. a user can only browse the type hierarchy of those entries to find entries; and (ii) no service is available to find and reuse PML provenance metadata published on the Web.

IWSearch is implemented as a service in the Inference Web architecture. It provides primitives for discovering, indexing, and searching for PML objects (i.e. instances of PML classes), such as pmlp:Person and pmlj:NodeSet, available on the Web. As shown in Figure 2, IWSearch consists of three groups of services:

1. the discovery services utilize swoogle [5] search results and a focused crawler (searching PML documents in a certain web directory) to discover URLs of PML documents throughout the Web;
2. the index services use the indexer to parse the submitted PML documents and prepare metadata about PML objects for future search, and uses the searcher to serve query calls invoked by users;

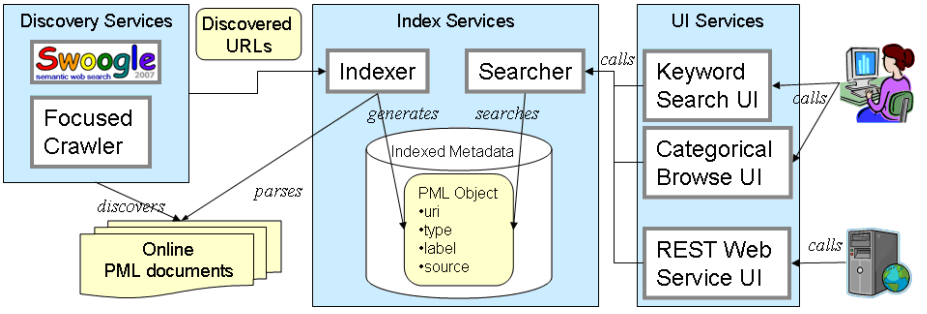

Fig. 2. IWSearch Architecture

3. the UI services offers a keyword search and categorical browsing interface for human or machine users.

On using IWSearch, one may reuse existing PML data that may be expensive to be created. For instance, PML data about a scientific publication may require the creator of the PML data to specify the publication's authors, authors' affiliations and publisher.

4.3 Probe-It!

Probe-It! is a browser suited to graphically rendering provenance information encoded in PML and associated with results coming from inference engines and workflows [3]. Probe-It! consists of four primary views to accommodate the different kinds of provenance information: queries, results, justifications, and provenance, which refer to user queries or requests, final and intermediate data, descriptions of the generation process (i.e., execution traces), and information about the sources respectively.

In a highly collaborative environment such as the cyberinfrastructure, there are often multiple applications published that provide the same or very similar function. A thorough integrative application may consider all the different ways it can generate and present results to users, placing the burden on users to discriminate between high quality and low quality results. This is no different from any question/answer application, including a typical search engine on the Web, which often uses multiple sources and presents thousands of answers back to users. The *query view* visually shows the links between application requests and results of that particular request. The request and each corresponding result is visualized as a node similar to the nodes in the justification view presented later.

Upon accessing one of the answer nodes in the *query view*, Probe-It! switches over to the justification view associated with that particular result. Because users are expected to compare and contrast between different answers in order to determine the *best* result, all views are accessible by a menu tab, allowing users to navigate back to the query view regardless of what view is active.

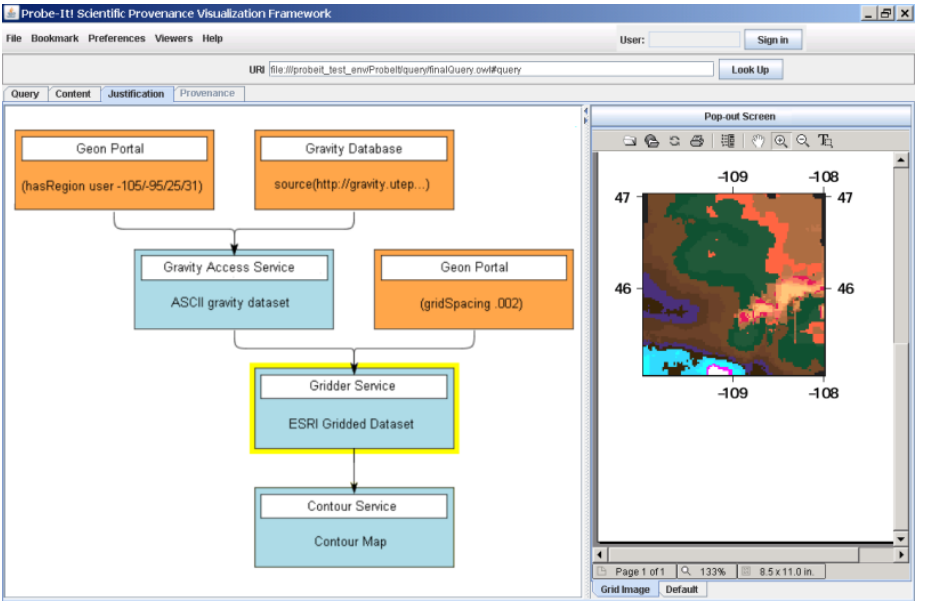

Fig. 3. Probe-It! justification view

The *results view* provides graphical renderings of the final and intermediate results associated with scientific workflows. This view is captured on the right hand side of Figure 3, which presents a visualization of a gridded dataset; this view is initiated by selecting one of the justification nodes, described in the next section. Because there are many different visualizations suited for gridded data and datasets in general, the *results view* is composed of a set of viewers, each implementing a different visualization technique suited to the data being viewed.

The *justification view*, on the other hand, is a complimentary view that contains all the process meta-information associated with the execution trace, such as the functions invoked by the workflow, and the sequencing associated with these invocations. Probe-It! renders this information as a directed acyclic graph (DAG). An example of a workflow execution DAG can be found on the left hand side of Figure 3, which presents the justification of a contour map. From this perspective, Web services and sources (i.e., data sinks) are presented as nodes. Nodes contain a label indicating the name of a source or invoked service, as well as a semantic description of the resulting output data. In the justification view, data flow between services is represented by edges of the DAG; the representation is such that data flows from the leaf nodes towards the root node of the DAG, which represents the final service executed in the workflow. Users can access both provenance meta-information and intermediate results of the sources and services represented by the DAG nodes. In this sense, the justification DAG serves as a medium between provenance meta-information and intermediate results.

The *provenance view*, provides information about sources and some usage information e.g., access time, during the execution of an application or workflow.

For example, upon accessing the node labeled *gravity database*, meta-information about the database, such as the contributing organizations, is displayed in another panel. Similarly, users can access information transformation nodes, and view information about used algorithms, or the hosting organization.

5 Evaluation of Lightweight PML

As MacEachren et al. describe in [8], *provenance* (or lineage, as mentioned in the reference) may be a requirement for understanding uncertainties related to geospatial information. In the case of our use case, map provenance is meta-information about source datasets, services and any other resource used to derive the maps [4]. In this section, we present some results of a user study used to verify whether provenance is needed for scientists to correctly identify and explain the quality of maps, a required condition if scientific communities are going to accept maps from CI-based applications. A comprehensive description of the user study can be found in [2]. It is important to note that this evaluation is part of a more comprehensive and ongoing effort to understand the need for provenance in scientific applications based on cyber-infrastructure resources. Also, the evaluation is not exactly about the lightweight use of the Inference Web. However, the results are significant for this particular paper because the provenance used in the study was *entirely encoded using lightweight PML*.

The user study analyzes how scientists with different levels of expertise on gravity data for geophysics and on GIS can differentiate between contour maps of high quality (e.g., maps with not known imperfections) and maps of low quality (e.g., maps with known imperfections) and to explain the reasons of identified qualities. Two cases *map* and *map+p* have been used, where only a single map M0 has been presented to subjects, thus no averages need to be taken; the scores are equal to the points earned for identifying and explaining the single map.

The hypothesis of concern for this paper is that "Scientists with access to provenance can identify and explain map imperfections more accurately than scientists without access to provenance." There are two types of scores associated with each evaluation case: an *identification score* and an *explanation score*, both of which have been used to assess the validity of our hypothesis. An identification score is computed as the average of points earned for correctly classifying the maps comprising an evaluation case. Similarly, the explanation score is computed as an average of points earned for correctly explaining the map imperfections. Because the measure of identifying and explaining maps is a binary value (e.g., 0 for incorrect answers and 1 for a correct answers), both types of evaluation scores are always between 0 and 1, inclusive.

Table 1. Subjects' average accuracy in identifying and explaining map imperfections

Task	*map*	*map+p*
Imperfection identification	0.10	0.79
Imperfection explanation	0.05	0.78

Significance of the the results collected so far were verified by a single-tail t-test at 95% confidence. The *imperfection identification task* in Table 1 contains the average accuracy of scientists in identifying maps quality. Condition cases *map* versus *map+p* tested whether provenance was needed in order to correctly assess maps; both cases are based on the same map containing the same error with the ability to access provenance in condition *map+p* being the only difference. Prior to the use of provenance, many scientists were unable to determine whether the map contained any imperfections at all, in which case their responses were regarded as unsuccessful earning 0. After the scientists were able to access the provenance, both their accuracy and confidence in determining the quality of the map improved significantly.

6 Discussion and Related Work

The uses of provenance are dictated by the goals of the particular systems; because various dimensions of provenance can be used to achieve various goals, there is no one use that fits all. For instance, one category of provenance systems aim at providing users with a sort of "history of changes" associated with some workflow, thus their view of provenance differs from that of a second category of provenance systems, which aim at providing provenance for use of debugging, or understanding an unexpected result. A third category of provenance systems record events that are well suited for re-executing the workflow it is derived from. From this point of view, PML fits into the second category of provenance systems. Provenance systems representative of these categories are reviewed below.

VisTrails, a provenance and data exploration system, provides an infrastructure for systematically capturing provenance related to the evolution of a workflow [6]. VisTrails users edit workflows while the system records the various modifications being applied. In the context of this system, provenance information refers to the modifications or history of changes made to particular workflow in order to derive a new workflow; modifications include, adding, deleting or replacing workflow processes. VisTrails provides a novel way to render this history of modifications. A treelike structure provides a representation for provenance where nodes represent a version of some workflow and edges represent the modification applied to a workflow. Upon accessing a particular node of the provenance tree, users of VisTrails are provided with a rendering of the scientific result which was generated as a result of the workflow associated with the node. In the context of VisTrails, only workflows that generate visualizations are targeted, however the authors describe how this system could be transformed to handle the general case as provided by PML in combination with Probe-It!; to provide a framework that can manage and graphically render any scientific result ranging from processed datasets to complex visualizations.

MyGrid, from the e-science initiative, tracks data and process provenance of workflow executions. Authors of MyGrid draw an analogy between the type of provenance they record for in-silico experiments and the kind of information that a scientist records in a notebook describing where, how and why experimental results were generated [15]. From these recordings, scientists are able

to operate in three basic modes: (i) debug, (ii) check validity, and (iii) update mode, which refer to situations when, a result is of low quality and the source of error must be identified, when a result is novel and must be verified for correctness, or when a workflow has been updated and its previous versions are requested. Based on particular user roles, the appropriate dimension of provenance is presented, knowledge, organization, data, or process level [15]. MyGrid is yet another system that supports different tasks or uses of provenance, thus there are multiple "modes" that users can operate in that effectively show only provenance relevant for a particular task. We believe that all levels of provenance are required in order for scientists to identify the quality of complex results.

All these provenance system thus far track provenance related to workflows. Trio is a management system for tracking data resident in a database; provenance is tracked as the data is projected and transformed by queries and operations respectively [14]. provenance related to some function is recorded in a lineage table with various fields such as the derivation-type, how-derived, and lineage-data. Because of the controlled and well understood nature of a database, lineage of some result can many times be derived from the result itself by applying an inversion of the operation that derived it. Additionally, Trio provides the capability of querying the lineage table, thus allowing users to request provenance on demand.

7 Conclusions

The Proof Markup Language has been used in several projects to encode application response justifications. While new justification requirements have been addressed by incremental enhancements of the PML specification, many PML-enabled applications and probably most future PML-enabled applications would need to use just a small subset of the PML concepts and concept properties. Furthermore, most users of these applications would need to use just a small set of tool functionalities to further understand application responses. Through the use of an exemplary scientific application, the paper was demonstrated that lightweight PML has been used by scientists to understand map imperfections. Moreover, the exemplary application has demonstrated the usefulness of IWSearch to support the reuse of PML metadata and of Probe-It! to support the visualization of knowledge provenance encoded with lightweight PML.

Acknowledgments: This work was supported in part by NSF grant HRD-0734825 and by DHS grant 2008-ST-062-000007.

References

1. Dean, M., Schreiber, G.: OWL web ontology language reference. Technical report, W3C (2004)
2. Del Rio, N., Pinheiro da Silva, P.: Identifying and Explaining Map Imperfections Through Knowledge Provenance Visualization. Technical Report UTEP-CS-07-43a, The University of Texas at El Paso (June 2007)

3. Del Rio, N., Pinheiro da Silva, P.: Probe-it! visualization support for provenance. In: Proceedings of the Second International Symposium on Visual Computing (ISVC 2), Lake Tahoe, NV, pp. 732–741. Springer, Heidelberg (2007)
4. Del Rio, N., Pinheiro da Silva, P., Gates, A.Q., Salayandia, L.: Semantic annotation of maps through knowledge provenance. In: Proceedings of the Second International Conference on Geospatial Semantics (GeoS), Mexico City, Mexico. LNCS, pp. 20–35. Springer, Heidelberg (2007)
5. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V.C., Sachs, J.: Swoogle: A search and metadata engine for the semantic web. In: Proceedings of the 13th CIKM (2004)
6. Freire, J., Silva, C.T., Callahan, S.P., Santos, E., Scheidegger, C.E., Vo, H.T.: Managing Rapidly-Evolving Scientific Workflows. In: Proceedings of the International Provenance and Annotation Workshop (IPAW) (to appear)
7. Ludaäscher, B., et al.: Scientific Workflow Management and the Kepler System. In: Concurrency and Computation: Practice & Experience (2005); Special Issue on Scientific Workflows
8. MacEachren, A., Robinson, A., Hopper, S., Gardner, S., Murray, R., Gahegan, M., Hetzler, E.: Visualizing Geospatial Information Uncertainty: What We Know and What We Need to Know. Cartography and Geographic Information Science 32(32), 139–160 (2005)
9. McGuinness, D.L., Pinheiro da Silva, P.: Infrastructure for Web Explanations. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 113–129. Springer, Heidelberg (2003)
10. McGuinness, D.L., Pinheiro da Silva, P.: Explaining Answers from the Semantic Web. Journal of Web Semantics 1(4), 397–413 (2004)
11. Murdock, J.W., McGuinness, D.L., Pinheiro da Silva, P., Welty, C., Ferrucci, D.: Explaining Conclusions from Diverse Knowledge Sources. In: Proceedings of the 5th International Semantic Web Conference (ISWC2006), Athens, GA, November 2006, pp. 861–872. Springer, Heidelberg (2006)
12. Pinheiro da Silva, P., McGuinness, D.L., Fikes, R.: A Proof Markup Language for Semantic Web Services. Information Systems 31(4-5), 381–395 (2006)
13. Pinheiro da Silva, P., McGuinness, D.L., McCool, R.: Knowledge Provenance Infrastructure. IEEE Data Engineering Bulletin 25(2), 179–227 (2003)
14. Widom, J.: Trio: A System for Integrated Management of Data, Accuracy, and Lineage. In: Proceedings of the Second Biennial Conference on Innovative Data Systems Research, Asilomar, CA, January 2005, pp. 262–276 (2005)
15. Zhao, J., Wroe, C., Goble, C.,, R.S.: Using Semantic Web Technologies for Representing E-science Provenance. In: Proceedings of the 3rd International Semantic Web Conference, pp. 92–106 (November 2004)