## First Workshop on the Theory and Practice of Provenance (TaPP '09)

*February 23, 2009*
*San Francisco, CA*

### INVITED TALK

- ***Causality, Responsibility, and Blame: A Structural-Model Approach***
  *Joe Halpern, Cornell University*

*Summarized by Kiran-Kumar Muniswamy-Reddy (kiran@eecs.harvard.edu) with assistance from Peter Macko*

Halpern's talk introduced the theoretical aspects of causality, responsibility, and blame and their implications for the provenance community.

The world can be modeled using structural equations that model the outcome of events using one or more random variables. The values of exogenous variables come from outside the model, while endogenous random variables depend on other variables in the system. There are, however, two choices of uncertainty over which reasonable people can disagree: first, we do not know whether our model is correct or sufficiently detailed, and, second, we may not be certain about the values of some variables. Consequently, we should introduce probabilities into the model that reflect our confidence in it. A model with no probabilities is completely deterministic.

Causality can be informally defined as follows. A set of events A caused event B, assuming that both A and B actually happened. Changing the outcome of A and possibly also the outcomes of some other events X would cause B not to happen. But if A happened, so would B, regardless of the outcomes of X. For example, this definition can gracefully handle the following scenario: it is sufficient to drop one match in order to burn a forest. If two people drop matches and the forest burns down, the definition correctly identifies both of them as the causes.

This definition of causality has one very important implication: it is not transitive. This challenges the interpretation of provenance as a form of causality, since provenance is believed to be transitive. The intransitivity of causality can be illustrated using the following example: a patient has an illness that can be cured by one dose of medicine, but two doses would kill him. On Monday, doctor A gives the patient a dose of medicine. On Tuesday, doctor B does not give him the medicine, because he already received it the day before. On Wednesday, the patient is alive and well. Clearly, doctor A caused doctor B not to administer the medicine, and by doing so, B caused the patient to be alive. If causality were transitive, A would also be a cause that the patient is alive, but this is not true: if doctor A did not give the medicine, the patient would still be alive on Wednesday, regardless of the subsequent action of doctor B.

While causality is binary, the degrees of responsibility and blame can be expressed as numbers between 0 and 1. Your responsibility is 0 if you are not a cause, or 1/(k+1), if k–1 is the minimum number of variables that must be changed in order to change the outcome of the event. For example, in a 6–5 voting scenario the responsibility of the six voters is 1, while in an 11–0 scenario the individual degrees of responsibility would be smaller. The degree of blame is the expected degree of responsibility given all possible situations considered by an agent. For example, consider the following situation: ten marksmen are ordered to shoot a prisoner. Nine of them receive fake bullets and only one gets a live bullet, but no one knows which one. If none of them misses the target, only the marksman with the live bullet is responsible with degree 1, while each of them has the same degree of blame 1/10.

Moving on to applications: causality can be used for model checking. If there is an error in the specification of a program, causality can be used to perform coverage estimation. In particular, one can ask the question, "Which parts of the spec cause the program to be satisfied?" If 90% is irrelevant, then the spec is flawed. Causal models can also be used when one is uncertain about provenance.

One member of the audience asked if the model the speaker presented considered the Trio definition of provenance. The speaker was unfamiliar with Trio, but it seems as though the two models address the same issues. For example, Trio tries to answer questions such as "Why does a tuple appear in a result?" which is basically causality.

## MORNING SESSIONS

*Summarized by Kiran-Kumar Muniswamy-Reddy (kiran@eecs.harvard.edu)*

- ***A Formal Model of Provenance in Distributed Systems***
  *Issam Souilah, University of Southampton, UK; Adrian Francalanza, University of Malta, Malta; Vladimiro Sassone, University of Southampton, UK*

In this talk, the speaker presented a provenance-based calculus to study trust in distributed systems, in particular, a formalism that is an extension of Pi-calculus. The basic idea is to annotate all data with their provenance. Users can use this provenance to make decisions; for example, a product made in China may be more desirable than a product made in Zimbabwe. They considered a few more approaches before deciding on this approach: static analysis does not scale. A dynamic analysis cannot do a full-blown verification or use proof-carrying code as decisions. Decision criteria need to be computationally lightweight. With their method, provenance tracking is automated and is orthogonal to programming. Their approach also ensures provenance annotation standardization and provides circular reasoning with respect to trust. One more contribution of the work is that they provide a definition for provenance correctness and prove the correctness for the provenance-tracking semantics that they have proposed.

One member of the audience asked what happens if the same value is sent to two different processes in two different channels. The speaker said that the messages sent on individual channels are considered to be different copies, so their approach holds. Next, James Cheney asked what happens if one of the entities lies and sends the wrong provenance. The speaker replied that they are assuming that the system is running in a trusted environment, i.e., the principals tell the truth about themselves so others can make decisions based on that.

- ***Towards Semantics for Provenance Security***
  *Stephen Chong, Harvard University*

*Summarized by Kiran-Kumar Muniswamy-Reddy (kiran@eecs.harvard.edu) with assistance from Peter Macko*

Stephen Chong presented formal definitions for provenance security that ensure that we do not reveal sensitive data and the provenance does not reveal sensitive information. The work assumes a simple language-based model, where the program has input locations and produces single output. The provenance trace T describes the execution of a program. The partial provenance of T allows parts of T to be elided (hide some values or even entire statements from T). At this point, an audience member asked who was deciding what provenance to elide. Stephen replied that the system assumes the existence of some access control scheme that basically tells whether some provenance should be elided. Each input location has a security policy for data and provenance (high or low security). A user knows the values of low-security inputs and is given output and partial provenance trace.

Basically, users are allowed to know the existence of objects but may not be allowed to know that they were involved in generating an output. In this system, T is secure if it accurately describes the execution and does not contain any high provenance locations, but that does not prevent reasoning using correlations. T satisfies provenance security if it is accurate and there exists another execution where the high provenance point is not even involved.

One member of the audience asked if there was a unique minimal reduction. The speaker replied that this was still an open question.

- ***Scalable Access Controls for Lineage***
  *Arnon Rosenthal, Len Seligman, Adriane Chapman, and Barbara Blaustein, The MITRE Corporation*

Adriane Chapman described securing the sensitive information that lineage contains. Role-based Access Control (RBAC), used in prior lineage security work, does not scale. RBAC might lead to situations where a new role has to be created for each user, leading to role explosion. Instead, Chapman proposed a model based on Attribute Based Access Control (ABAC). In classic ABAC, attributes are used to determine access. If the attributes, such as age or taste, satisfy some properties, then access is allowed. However, ABAC rules are hard to change once specified. Instead, they allow stakeholders to specify the access allowed, and when the stakeholders have conflicting opinions, they reconcile them. In their system, the basic ownership defaults are that the process node is owned by the process creator, the data node is owned by the data creator, and the edge between nodes is the union of stakeholders. They have further extended ABAC to return a fake node if a user is denied access to a node.

Margo Seltzer pointed out that not just the nodes of the provenance graph but even the edges connecting the graph have provenance. Chapman replied that taking the union of the permissions at the edges is sufficient. Next, Erez Zadok asked if they had considered the MLS model, to which Chapman replied that they had not thought about it. Another member of the audience asked if giving away a bogus node isn't giving away some information. Chapman replied that they are still working on this aspect. Another member asked if the fact that some entities have the ability to create nodes does not leak information. Chapman replied that they are looking into it.

- ***On Explicit Provenance Management in RDF/S Graphs***
  *P. Pediaditis, G. Flouris, I. Fundulaki, and V. Christophides, ICS-FORTH*

The context of the work is provenance management in RDF/S (a collection of data and schema triples). RDF/S (Resource Description Framework Schema language) is used to add semantics to RDF triples by imposing inference rules. This entails new implicit triples (facts) that are not explicitly asserted. Deletion/update of some triples will ensure that some of the implicit information will be lost, even though some of that information is still valid. To solve this, they introduce RDF/S graphs that help these issues by performing queries and updates.

RDF/S graphs are a set of RDF named graphs that are associated with a URI and have a set of triples whose ownership is shared by the named graphs that constitute the graph set. The authors further extended RQL to handle provenance queries and to support updates to graph sets through an extended version of RUL.

An audience member said that if things change, inference might change, so why should we care about retaining it? The speaker replied that this is because they are using coherent values and not operational semantics. Another member of the audience asked if they had studied whether querying semantics could be simulated with foundational or operational semantics. The speaker replied that there has been a lot of study in the literature, but there is no clear marking between the two.

- ***Application of Named Graphs Towards Custom Provenance Views***
  *Tara Gibson, Karen Schuchardt, and Eric Stephan, Pacific Northwest National Laboratory*

Tara Gibson described making provenance more accessible to users. Workflow provenance is very detailed and presents a machine view of things, which is probably much more detailed than a human can understand. In this talk, Tara presented a filtering technique that extends SPARQL (SPARQL Protocol and RDF Query Language) to help avoid information overload. Furthermore, since this is a generic extension to the query language, users do not have to constantly rewrite code every time they want to customize their views.

In particular, their approach leverages extensions to RDF Named Graphs (NG). They extend SPARQL to include the new keyword APPLY, which tells the query interface what views should be applied to the query result. Based on the views specified, each NG is grouped into a user-defined node. The new node is then created based on the properties associated with the NG definition. They then restore links between the new aggregate nodes in the result. Currently, they are working on implementation of the system.

An audience member asked if they know of any properties that must hold for their extension to transform the nodes and whether their method destroys some of these properties. The speaker replied that since they still maintain the original graph at the lower level, users can get back to it if they want to. Another member from the audience asked if they allow users to abstract things at runtime. The speaker replied that in theory it is possible.

- ***Authenticity and Provenance in Long Term Digital Preservation: Modeling and Implementation in Preservation Aware Storage***
  *Michael Factor, Ealan Henis, Dalit Naor, Simona Rabinovici-Cohen, Petra Reshef, and Shahar Ronen, IBM Research Lab in Haifa, Israel; Giovanni Michetti and Maria Guercio, University of Urbino, Italy*

Michael Factor discussed his team's work on long-term digital preservation, which involves processes, strategies, and tools to allow future usability of digital assets. Their work leverages the Open Archival Information System (OAIS) standard for digital preservation. It consists of content information (data and representation information) and preservation descriptive information (reference, provenance, context, fixity, and representation information).

Architecting a preservation store that supports authenticity and provenance is a challenge. To this end, their work (and the talk) presented a novel model for managing authenticity in a preservation environment and an implementation that integrates the concept of provenance.

Someone asked if the OAIS standard makes it easy to preserve information. The speaker replied that their work provides a concrete implementation of the model. Another audience member asked if there are scenarios where a bit stream can change without semantics also changing. The speaker replied that an example of this is when you change the format of a document from MS Office 97 to Office 2000. The challenge is to ensure that data has not been corrupted during the migration from one format to another. A final questioner pointed out that many archival experts do not consider format conversion an appropriate mechanism and prefer the use of virtual machines. The speaker replied that some data is application-independent, and that does need to be updated as systems are updated.

- ***Steps Toward Managing Lineage Metadata in Grid Clusters***
  *Ashish Gehani and Minyoung Kim, SRI International; Jian Zhang, Louisiana State University*

Ashish Gehani explained that their work is set in the context of the grid, i.e., distributed systems with non-interactive workloads that involve a large number of files. The goal is to provide low-latency lineage queries that enable a number of applications: for example, dynamic toolchain selection, safety/reliability (check tool dependencies), etc. The speaker then presented a set of discarded approaches which included use of auxiliary files, a local database, in-band encoding, and headers and footers to store provenance, as well as making the file server provenance-aware.

They are currently experimenting with a hybrid approach which uses an overloaded namespace for storing provenance. In this approach, the system sends provenance when the end of a file is reached. This approach transparently makes protocols such as FTP and SCP provenance-aware. They also built a scheme where lineage is replicated at the nodes that actually consume it, thus ensuring that lineage is more readily available at those nodes. Finally, they store all the lineage in a HyperTable distributed database, ensuring that all clients have access to the lineage.

An audience member asked why they don't store all the provenance in a central location and then just query it. The speaker replied that as long as the users do not care about synchronous queries, the central-location approach should not be a problem.

## INVITED TALK

- ***The State of Provenance in 2019***
  *Margo Seltzer, Harvard University*

*Summarized by Peter Macko (pmacko@fas.harvard.edu)*

It is the year 2019, and we won! Provenance is everywhere. It is secure, reliable, mandatory, and globally searchable. All kinds of storage systems collect provenance, including local file systems, network-attached storage, and storage in the cloud. All major programming languages are provenance-aware; consequently, all programs implemented in them collect provenance. Web browsers are provenance-aware, and so is the entire Web.

Provenance is secure and verifiable, and users can easily restrict access to the provenance of their objects in order to prevent release of confidential information. For example, if we had this technology back in 2009, the public would never learn the value of the Facebook settlement. Provenance collection cannot be turned off, which makes forging digital signatures or plagiarism easily detectable! All viruses can be tracked immediately back to the Web sites they were downloaded from, which helped to solve the virus problem.

How did we get there? Back in 2004, efficient provenance collection was not even thought to be possible. Soon, the first few provenance-aware (PA) systems appeared: domain-specific solutions, workflows, and storage systems. The next big step was the development of PA programming languages, which introduced more detail into provenance collection and essentially eliminated false provenance. Then the first few PA applications were developed, which was soon followed by the emergence of PA protocols. Most remarkably, email provenance solved the spam problem. And, starting in 2010, there was some fundamental work on graph databases and formalism, which greatly improved querying provenance graphs.

Fortunately, space and computational overhead of provenance collection simply became irrelevant during the past 10 years. The cost of storage went down, so that a terabyte is essentially free. Processors have hundreds of cores, and we still do not know how to exploit all this parallelism. You can devote a hundred cores solely to provenance collection and no one would ever notice any performance impact.

Formalism was also necessary for the success of provenance. It was discovered that causality is not transitive, which forced us to rethink what provenance really is. The community then approached provenance from the perspective of an information flow, and soon provenance semantics and calculus were developed. Related security work solved issues of the reliability and verifiability of provenance, which helped provenance become accepted as the method for system auditing. Another important aspect of security is protecting provenance itself in order to prevent leakage of confidential information.

The success of provenance would not be possible without proper standardization (in order to allow interoperability) and policies. Provenance was standardized through a grassroots effort, creating a need for provenance before standardizing it, and instead of inventing the standard, standardizing the practice. The need for provenance was created by the financial crisis of 2009: In 2010, the government passed an act that required companies to prove additional regulatory compliance, and provenance was perfectly suited for this purpose. The provenance community jumped on this bandwagon and was soon joined by industry.

This talk spawned a lot of interest and started a long discussion. One member of the audience reminded the speaker of a thriving black market of legacy systems without provenance, while another talked about "provenance-free Finland." When asked about the people who do not care about provenance, the speaker explained that they would not be affected in any way—provenance collection is transparent, essentially free, and comes with reasonable security defaults.

Other members of the audience were concerned with the Big Brother aspect of provenance and the (in)ability to post anonymously on the Internet. The speaker agreed that provenance has the potential for misuse, but this risk can be made minimal if the security settings are correctly configured. However, there would always be new ways to find exploits. Other issues raised by the audience included topics such as pre-existing unannotated data, false positives, and implicit information flows.

## FIRST AFTERNOON SESSION

*Summarized by Peter Macko (pmacko@fas.harvard.edu)*

- ***A Framework for Fine-grained Data Integration and Curation, with Provenance, in a Dataspace***
  *David W. Archer, Lois M.L. Delcambre, and David Maier, Portland State University*

Some tasks require fine-grained data integration from a large variety of sources, which is done manually by experts in the particular field. This process usually involves copying and pasting data from database tables, emails, documents, and Web pages, followed by manual editing and cleaning. The talk presented the research in capturing the provenance of such process. The collected information can be used to answer questions such as: where does the data in this column come from? if there are multiple conflicting data sources, which of them was preferred by the user?

The research group developed a provenance-aware editor of tabular data, which logs all user actions. When data is copied and pasted from an external application, the editor annotates it with information about where the data came from. This information is currently provided by Microsoft applications via an Office add-in, or it can be entered manually. The editor explicitly supports entity resolution, in which the user merges two rows that correspond to the same entity (event) and resolves any conflicting values. Similarly, the user can resolve two columns that correspond to the same attribute.

The system can parse the edit log and produce a provenance graph for any given data value. The graph shows where the value came from, when it was entered, and, if the user performed any form of resolution, what the conflicting values were. The log can also be queried to learn a wide variety of information about both individual values and sets of values, such as the names of all data sources, the age of the oldest source, or the total number of resolutions.

- ***The Case for Browser Provenance***
  *Daniel W. Margo and Margo Seltzer, Harvard University*

Web browsers keep track of a large amount of information, which causes "a little big data management problem." The amount of data is tractable for a computer but not for users. The traditional solution is bookmarks. More advanced solutions include auto-complete, history search, and the smart location bar. These features are based on history and usage statistics, which is, in fact, provenance. In this talk, Daniel Margo explored how this provenance, which is already collected by the browser, can be used to provide additional sophisticated features.

One of the possible use cases is determining the lineage of downloaded files, because in many cases the URL alone is not sufficient. For example, learning that a picture was downloaded from ImageShack is usually not good enough. Instead, browsing history can be used to determine the exact sequence of user actions to obtain the given downloaded file. Provenance can also be used to improve history search and personalize Web search. For example, the browser can use it to learn that when a user searches for "rosebud," she is interested in gardening, but not in *Citizen Kane*. The browser can use this extra information to refine Web search results or clarify the search query by adding the word "flower" to the query.

This talk was followed by a long discussion—almost as long as the talk itself. Some members of the audience were concerned about security: what if someone steals the collected provenance? what if someone modifies it? The speaker pointed out that browsers already collect all the data (they are just not using it), so if this is the concern, we should already have it now. He also explained that we do not need to worry about users faking their provenance, because, in the end, it would only hurt them. When asked about how much history is necessary for these features, the speaker admitted that he did not study this issue. He mentioned that his personal browsing history was sufficient to get reasonable results.

Other audience members were concerned about the irrelevant parts of the history, such as clicking on an uninteresting link or the possibility of including irrelevant steps in a lineage of a downloaded file. The speaker explained that uninteresting links are being handled gracefully by the applied graph algorithms. When tracing back through the download lineage, most users can identify parts that they recognize before reaching the earlier irrelevant actions.

- ***Provenance as Data Mining: Combining File System Metadata with Content Analysis***
  *Vinay Deolalikar and Hernan Laffitte, Hewlett Packard Labs*

A large amount of information is stored in an unstructured setting as (text) documents without provenance. This talk presented a method of discovering the provenance of a document by combining file system metadata with data mining techniques. The advantage of this approach is that it does not assume any modification of the file system; consequently, it can also operate on legacy documents.

The algorithm is based on the following two observations: two documents that are one link apart in a provenance chain tend to be similar, and the direction of the information flow can be inferred from the file creation and modification times. The algorithm starts by coarsely clustering documents by their feature vectors, such as TF-IDF (term frequency-inverse document frequency). It then identifies the cluster that contains the document of interest, reclusters it finely, and adds other related documents (such as those in the same directory) to the working set. Then, starting at the document of interest, the algorithm works backward by finding files with similar feature vectors until some stopping criterion is met.

The experimental results presented by the speaker showed that this approach is indeed effective and produces few false positives, but many members of the audience were skeptical. The speaker clarified that this algorithm works even if you do not keep old versions of your documents, although many users do indeed keep some. Other concerns raised by the audience included scalability, reproducibility of results, and application of this technique to other domains, such as images or media.

## FINAL SESSION

*Summarized by Richard P. Spillane (necro351@gmail.com)*

- ***Story Book: An Efficient Extensible Provenance Framework***
  *R. Spillane, Stony Brook University; R. Sears, University of California, Berkeley; C. Yalamanchili, S. Gaikwad, M. Chinni, and E. Zadok, Stony Brook University*

Richard Spillane argued that provenance-aware systems should make it easier to support application-specific provenance tracking and should at the same time utilize a simpler design. Simply logging provenance is a straightforward design that leads to a more stable and reliable implementation, but it consumes too much disk space. Spillane introduced Story Book, a system that utilizes a write-optimized database and a FUSE-based file system to log provenance records and general compression algorithms to reduce disk consumption. Rather than maintain a provenance graph in memory while capturing provenance events, these events are instead logged and compressed. Later, during queries, a provenance graph is constructed from the indexed log records. To support application-specific modifications, Story Book allows developers to record different provenance for different file types upon file system access. Story Book also has an API to allow applications to manually insert application-specific provenance.

To evaluate the performance of Story Book, Spillane compared Story Book to Waldo, the log indexing tool implemented in PASS (Muniswamy-Reddy et al., *2006 USENIX Annual Technical Conference*), and also implemented an alternative version of Story Book that stores provenance log records in a traditional read-optimized database (Berkeley DB). Spillane noted that Story Book performs as expected: its write-optimized version is 2.8 times faster than its read-optimized version when processing provenance log records. In comparison, Waldo performs 3.5 times faster than Story Book's write-optimized version. Waldo, however, was inserting pre-compressed data while Story Book was inserting uncompressed data; Waldo is a subset of PASS, which performs its compression before log indexing. Spillane noted that read performance for write-optimized Story Book is 3 times slower than for read-optimized Story Book, but argues that this is an unimportant workload for provenance tracking, which he asserts is generally a logging workload.

One audience member asked whether the authors had considered providing multiple alternative provenance graphs to users on queries. Spillane replied that they had not but that Story Book's design did not preclude such a feature. Another member asked whether Story Book provided provenance graphs on queries. Spillane asserted that a full and complete provenance graph is provided on queries.

- ***Making a Cloud Provenance-Aware***
  *Kiran-Kumar Muniswamy-Reddy, Peter Macko, and Margo Seltzer, Harvard University*

Kiran-Kumar Muniswamy-Reddy outlined the motivation for a provenance-aware storage system (PASS) which used a remote database (Web service) as a backing store. He argued that storage sizes for large scientific data sets are becoming unwieldy and that using a shared storage back end is a practical way to share storage costs among multiple labs. If, however, some lab still wants to track the provenance of files accessed by all labs, a PASS built on top of the Web service is required. Muniswamy-Reddy clarified, however, that creating a PASS on top of a Web service is complicated by the differing semantics of different Web service providers. He argued that a practical PASS should satisfy: (1) a Read Correctness constraint, which dictates that reads on data should be accompanied with up-to-date provenance; (2) Causal Ordering, which dictates that the provenance of a file-update or process action is always complete as of the time that operation occurred; and (3) Efficient Query, which specifies that provenance queries should be feasible. Muniswamy-Reddy outlined the evolution of a PASS built on top of a Web service that eventually satisfies all these constraints.

Muniswamy-Reddy described three systems: (1) S3, an object store that allows the insertion and removal of key and value pairs; (2) SimpleDB, which supports at least insertion into and query on a basic table with 256 attributes; and (3) SQS, which acts as a basic queuing service for clients to pass messages through. He first described a system that uses only S3 to store both data and the provenance of this data but is limited in query performance, violating the Efficient Query constraint. He improves query performance by storing provenance data in SimpleDB and data in S3; however, this violates the Read Correctness constraint. Muniswamy-Reddy's final design utilizes the work of Branthner et al. and uses SQS as write-ahead logging order to ensure Read Correctness. The final design incurs an estimated 32.2% overhead on top of simply copying the data (without storing provenance) to a remote Web service. Muniswamy-Reddy estimates that roughly 71,000 operations on the remote Web service would be required to recover the provenance of a file in his benchmark data set, which he asserts is reasonable.

One audience member asked what kinds of modifications Muniswamy-Reddy would make to one of the Web services he utilized in order to better support provenance capture. He replied that he would modify the service to capture and store the provenance internally, without involving the client.

- ***Transparently Gathering Provenance with Provenance Aware Condor***
  *Christine F. Reilly and Jeffrey F. Naughton, University of Wisconsin, Madison*

Christine F. Reilly described the modifications she made to Condor, a distributed job executing environment, to track the provenance of files modified by jobs. Condor by itself does not do this, but Reilly utilizes two new extensions to extend Condor to track provenance: Quill, a tool to scan and insert Condor logs into an SQL database, and FileTrace, a tool to track interactions between jobs and a shared file system. This extended system is called Provenance-Aware Condor, or PAC. The primary advantage of PAC is that it requires no modification to the source code of any job that could run in Condor.

To evaluate PAC, Reilly extrapolated the size of a provenance table for jobs that ran for a year from shorter-running jobs. She estimated that the largest provenance table would be 1.15TiB (tebibyte) after a year of running, and the second largest table would be 0.9TiB. Reilly evaluated four synthetic applications in PAC and in an unmodified Condor system (no Quill or FileTrace extensions). The paper shows the overheads for these synthetic workloads as negligible. Finally, Reilly looked at the application of PAC to tracking the provenance of a Web site called DBLife, which is an online community designed to manage information about the database research community. Limitations of PAC were illustrated in this section of the talk—namely, the lack of application-specific knowledge needed for answering DBLife-specific provenance queries.

One questioner asked about the overhead of the FileTrace tool itself on top of an untracked process. Reilly had not yet investigated this. Some audience members commented that despite the small amount of file data modified by a job, the estimated size of the provenance table after a year was indeed large, contrary to Reilly's claim. When asked about how PAC tracks processes that are being waited upon by parents or are already being tracked, Reilly responded that such applications weren't typically run on a Condor system.