Implementing interoperable provenance in biomedical research

V. Curcin[a,*], S. Miles[b], R. Danger[a], Y. Chen[b], R. Bache[b], A. Taweel[b]

[a]*Department of Computing, Imperial College London, London SW7 2AZ, United Kingdom*
[b]*Department of Informatics, Kings College London, Strand, London WC2R 2LS, United Kingdom*

Abstract

The provenance of a piece of data refers to knowledge about its origin, in terms of entities and actors involved in its creation, e.g. data sources used, operations carried out on them, and users enacting those operations. Provenance is used to better understand the data and the context of its production, and to assess its reliability, by asserting whether correct procedures were followed. Providing evidence for validating research is of particular importance in the biomedical domain, where the strength of the results depends on the data sources and processes used. In recent times, previously manual processes have become fully or semi- automated, e.g. clinical trial recruitment, epidemiological studies, diagnosis making. The latter is typically achieved through interactions of heterogeneous software systems in multiple settings (hospitals, clinics, academic and industrial research organisations). Provenance traces of these software need to be integrated in a consistent and meaningful manner, but since these software systems rarely share a common platform, the provenance interoperability between them has to be achieved on the level of conceptual models. It is a non-trivial matter to determine where to start in making a biomedical software system provenance-aware. In this paper, we specify recommendations to developers on how to approach provenance modelling, capture, security, storage and querying, based on our expe-

*Corresponding author

Email addresses: vasa.curcin@imperial.ac.uk (V. Curcin), simon.miles@kcl.ac.uk (S. Miles), r.danger@imperial.ac.uk (R. Danger), yuhui.chen@kcl.ac.uk (Y. Chen), richard.bache@kcl.ac.uk (R. Bache), adel.taweel@kcl.ac.uk (A. Taweel)

riences with two large-scale biomedical research projects: Translational Research and Patient Safety in Europe (TRANSFoRm) and Electronic Health Records for Clinical Research (EHR4CR). While illustrated with concrete issues encountered, the recommendations are sufficiently high level so as to be reusable across the biomedical domain.

Keywords: provenance, biomedical informatics

---

1. Introduction

Provenance aims to capture the origin of some data through details of actions and actors involved in its creation. In scientific applications, provenance helps us to understand research results [1]. For instance, a published clinical study may contain a table showing the statistical significance of some treatment on the case group, as opposed to the control sample. Provenance of that table would consist of the statistical algorithms used, their parameterisation, data cleaning that was applied, case and control definitions, and information about the data provider or the data gathering process used. In some circumstances, a part of the process may change over time (e.g. tweaking the case definition) causing the result to change, and the provenance trace can provide clear information about how the result was obtained and how it may be repeated or improved.

Provenance is directly contributing to several important goals that research methodologies are trying to attain. In itself, provenance traces make the research process *auditable*, by providing a standardised account of actions that unfolded during the process execution. Combined with a formal model, such as a business workflow specification, provenance ensures the results are *verifiable*. Finally, when the program executables are provided together with the data used, they jointly ensure *reproducibility* of the research.

Biomedical research is characterised by the heterogeneity of the research teams participating in projects, procedures they follow, and the data they produce. A drug development pipeline would span a range of disciplines from target identification via detection of candidate genes for drugs, to clinical studies exploring the efficacy and drug safety. The need to capture details of data produced at each step and the processes involved is persistent throughout this process and benefits from common technical frameworks that span different scientific domains and multiple teams. For example, collaborative workflows [2] have proven to be highly useful in integrating microarray

analysis with low-level gene annotation.

Auditability and verifiability of research data are also essential components of data management in clinical research, due to the sensitivity and importance of its impact on saving lives. This is reflected in popular standards such as GxP (including Good Clinical Data Management Practice and Good Clinical Practice) [3], CONSORT for trial reporting [4], and STROBE [5] for reporting observational studies. Of particular interest is ADAM [6] produced by the Clinical Data Interchange Standards Consortium (CDISC), which documents each derived variable (treatment, outcome, or covariate) used in clinical trial analysis data sets, to enable review and re-creation of published research. All of these standards take a retrospective view of data provenance, as something that needs to be collected and described post-hoc. As will be shown, successful provenance implementations adopt a prospective view, automatically collecting this information in a single repository during the life of a research project.

Reproducibility is also the focus of The Open Data initiative [7] which aims to make publicly generated data free and available to everyone, in useful formats, subject to proper attribution. Another part of that vision, directly relevant to health data management, is that any published research study should be accompanied by the full data that it was derived from, thus enabling the reader to verify the results for themselves. This approach is increasingly taken up by scientific journals [8].

In this paper, we review the implications of provenance for biomedical research by analysing the provenance requirements of two real-world use cases from the clinical research domain, and propose recommendations on appropriate solutions for developing provenance capacity. In particular, we chose the use cases that rely on the service oriented architecture paradigm to highlight the importance of provenance in a complex computing system and reveal the benefits that the provenance capacity may bring.

2. Background

The concept of provenance is well established in many disciplines [9, 10]. For example, in the study of fine art it refers to the trusted, documented history of some work of art. Electronic tracking of provenance was originally studied in individual domains including geography or library studies, or with regards to particular technologies, such as databases or workflow systems. It was recognised that the same issues occurred in these different

applications, and so similar solutions may apply. Simultaneously, there was a push from many organisations and projects for a standard approach to representing provenance, as this would then allow systems to be developed with some guarantee that the provenance data held would be interpretable in the future. Furthermore, it was understood that an important effect of having common provenance representations would be that the history of data could be traced across multiple heterogeneous systems, as the provenance each system recorded would be interoperable and interconnectable with that recorded by the others.

2.1. Provenance representation models

In the early days of the provenance efforts, several generic provenance models were proposed [11, 12, 13]. Several metadata vocabularies also allowed some limited provenance information to be expressed, particularly Dublin Core [14] or Minimum Information About a Microarray Experiment (MIAME) [15] for gene expression data.

Through merging the pioneering efforts of several research groups, a community-driven provenance specification, the Open Provenance Model (OPM) [16] became a de-facto standard representation for provenance in many areas. OPM is a causal graph model, with edges denoting relationships (X was caused by Y) and nodes representing the individual occurrences of entities. The OPM type graph is shown on the left side of Figure 1. OPM graphs describe the full lineage of a piece of data in terms of multiple events (process instances) that led to it being produced. The nodes in the graph can be of three types: artifacts, processes, and agents.

- Artifacts are pieces of data of fixed value and context, e.g. one version of a data set, or a document, and are denoted by ovals in the provenance graph.

- Processes are actions that are performed using artifacts to generate other artifacts, e.g. a selection process uses the full set of eligible patients and generates a subset of these with which to conduct the trial. They are represented by rectangles in a provenance graph.

- Agents are the entities controlling process execution, either as human actors or as non-mutable pieces of software. They are shown as octagons in the provenance graph.

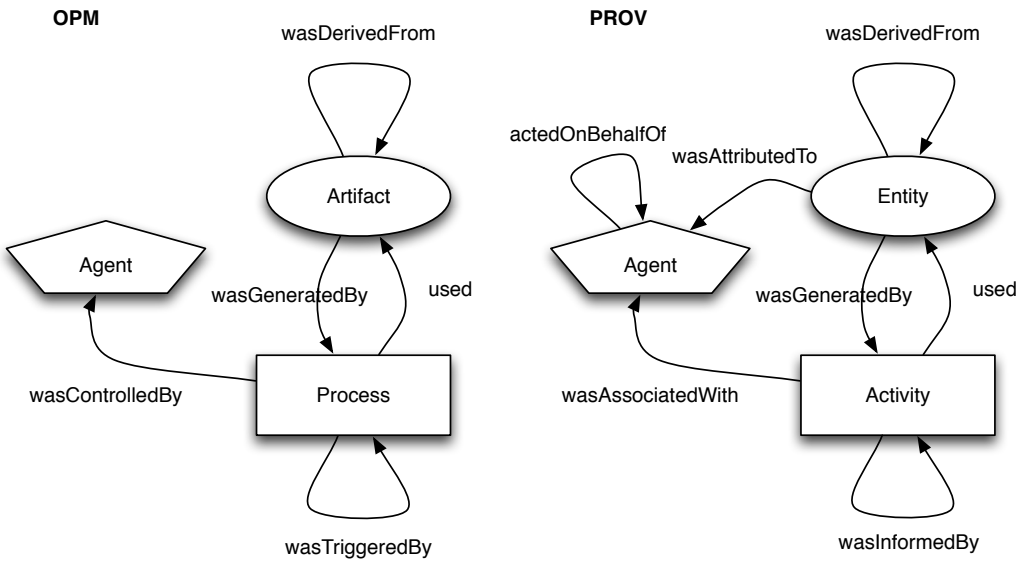

Figure 1: OPM and PROV type graphs showing available kinds of nodes and edges in each language

The various properties of artifacts, processes and agents are represented by arbitrary key-value annotations to the nodes. These annotations can include any contextual information, such as regarding the type, ownership, data format, etc. of the data or activities represented by the nodes. Edges can also have annotations to provide further information on how one occurrence caused another.

Building on that and other work, the W3C standards body initiated a working group to develop official W3C standards for provenance. The PROV family of models [17], the type graph for which is shown on the right side of Figure 1 is based on OPM, revising and extending it with better support for attribution and evolution of entities over time. PROV activities denote something that occurs over a period of time, and they both use and generate PROV entities. PROV entities are things, whether data, physical or conceptual. Unlike OPM artifacts, entities can be mutable and have a lifetime between being generated and being invalidated (by activities). One entity can be a specialisation of another entity in a particular context, e.g. a patient's medical record with particular contents following a consultation is a specialisation of that medical record as it has existed over time. PROV agents denote that which has responsibility for something, commonly a person, or-

ganisation or software agent. They may be responsible for an activity having taken place, or for an entity existing. The latter can be used to attribute authorship of some data to a person, for example. PROV also provides basic modelling infrastructure for describing data structures in the provenance, through the use of collections and dictionaries. The nodes and edges of a PROV graph can have attributes, which are the data defining their value and context, e.g. the number of patients identified for a clinical trial or the title of a report.

This type of provenance is sometimes referred to as *retrospective* provenance, in that it captures historical data produced by execution of some process model. Nomenclature used in [18] and [19] distinguishes *prospective* provenance which is the representation of the process model that creates retrospective provenance data, and the *process* provenance which looks into the provenance of that process model itself. This categorisation is particularly well-suited to systems where there is a single computational artifact producing all the provenance data, such as a scientific workflow in a workflow management system. Then the prospective provenance is captured by the workflow being used, and the process provenance is the provenance of that workflow as it evolves over time.

When provenance is represented by a graph model, such as OPM and PROV, the queries are usually performed using a graph query language with semantic extensions to integrate with the domain of interest. Graph query languages have been extensively investigated for applications such as semi-structured data, semantic graphs, transportation networks, social networks and others. A good overview can be found in [20]. The typical queries in such languages focus on subgraph matching, finding nodes connected by paths, comparing and returning paths, aggregation, node creation, and approximate matching and ranking. Early languages include **G**[21] and GraphLog [22], an extension of DataLog. Object-oriented graph models were introduced in languages such as GraphDB [23], GOOD[24], Lorel[25], and Strudel[26]. UnQL [27] uses structural recursion and a functional model to work on semi-structured data. YAGO/NAGA[28] was developed for semantic search engines, and contains weighting elements that are used to represent the confidence of information in the query result. These languages influenced the semantics of the SPARQL language for querying RDF graphs [29], e.g. v1.1 of the standard included the capability to query paths using regular expressions. With RDF commonly used for persisting provenance data, SPARQL is frequently used as the language for querying provenance.

2.2. Provenance of biomedical software

In biomedical research, the data (genotype and phenotype information from multiple sources), the workflow (procedures carried out to perform the data analysis) and the log records (recording of relevant events in those procedures) may be distributed among several heterogeneous and autonomous information systems [30]. These information systems may be under the authority of different healthcare actors such as general practitioners, hospitals, hospital departments, etc, which form disconnected islands of information. Provenance frameworks then take the additional role of verifying the adherence of actors in the process to security policies in place.

The provenance of many pieces of information, such as a count of patients eligible for a clinical trial, is derived from data in multiple sites. Knowing the provenance of the results means having records of what occurred at each of those sites, and integrating these. In order for this to happen, each site's software must generate records that can be integrated, i.e. that are interoperable, ideally by using the same base model.

When the tools share the same execution environment (workflow management system, scripting engine etc.) this model can be based on the shared framework [31, 32], but this is not an option in the presence of heterogeneous software. Similarly affected is the prospective provenance aspect, which cannot rely on describing a scientific workflow or another concrete process that is generating provenance, but has to look into more abstract structures that are shared between multiple software systems. Thus, interoperability is the primary driver for such model-based approach to provenance.

Just because the full details of all processes in the system, and the data they accessed, could be documented and made accessible to queriers, this does not mean that any particular piece of provenance data should be communicated outside of a given site or that any particular individual should have access to all provenance. The secure provenance problem [33] is the task of providing assurances of integrity, confidentiality, and availability to the tasks and provenance records. Preserving the *confidentiality* of provenance data and verifiable application of security policies associated with it, is of particular concern to biomedical research.

The biomedical domain includes a plethora of data models and ontologies, allowing distributed applications to share common terminology. Vast majority of terminologies, formats, and standards in use are publicly available and managed, practice stemming from the need for public and governmental scrutiny of medical guidelines. Thus, commercial organisations focus

their business model around the data rather than proprietary formats that would lock users into their software tools. For example, Clinical Data Interchange Standards Consortium, comprised of industrial, clinical, and research organisations, is tasked by the US government to promote standards and interoperability in the biomedical domain. Thus, as well as integrating the recording and storage of provenance information into the existing software of a distributed application, the model used for provenance needs to express concepts specific to the biomedical domain, and so be integrated with those vocabularies and models.

Ultimately, the value of recorded provenance data in biomedical domain is that it provides answers to questions about the research conducted. However, once the users start working with provenance data, they may think of new provenance questions they could answer. Therefore, the way in which provenance is modelled and recorded needs to be *future-proof*, as far as is feasible, in being able to answer provenance questions not considered at design time.

In summary, the way in which provenance is modelled and recorded in a biomedical research application needs to allow interoperability of provenance between subsystems, provide confidentiality over portions of the provenance data, be able to be integrated with domain models, and be future proof with respect to new provenance questions that may arise.

3. Use cases

In this section we give a brief overview of the two research projects, experiences of which motivate our general recommendations that follow. The two projects, TRANSFoRm (Translational Research and Patient Safety in Europe) [34] and EHR4CR (Electronic Health Records For Clinical Research) [35] are investigating the integration of data required for clinical trials and data routinely collected in medical practice.

Clinical trials are an important part of medical research and new drug development. In a biomedical project, upon the satisfaction of the pre-clinical trials, a set of tests have to be conducted with patients with specific health conditions to generate safety and efficacy data. The number of patients recruited for trials has to be sufficient so that the experiment results will be statistically significant for examining the effectiveness of a new treatment. However, recruiting patients with the predetermined characteristics for a clinical trial is a time-consuming and costly process. Traditionally, clinical

researchers have to communicate with a number of healthcare centres in order to find eligible patients and suitable sites for conducting clinical trials. They then have to go on to negotiate with individual centres in order to execute the clinical trial with the eligible patients at each centre.

TRANSFoRm focuses on general practice and makes use of large collections of data collected therein, while EHR4CR is more concerned with the hospital setting and integration of data from a larger number of specialist data sources.

3.1. TRANSFoRm (Translational Research and Patient Safety in Europe)

The TRANSFoRm project is developing a common digital infrastructure to support the vision of the learning health care system [36] through integrating the data and workflows of clinical and research domains in primary care. The project outputs include methods, models, services, validated architectures and clinical demonstrations of software to support this integration. The software has been designed as a modular collection of tools that are deployed in three software configurations: Epidemiological Study, Randomized Clinical Trial, and Decision Support. These configurations directly support the three use cases in the project: retrospective diabetes cohort study, gastroesophageal reflux disease clinical trial, and a diagnostic support system for chest pain, abdominal pain, and dyspnoea.

At the heart of the Epidemiological Study software configuration is a semantically aware Query Formulation Workbench, designed to enable easy authoring of distributed searches to EHR and other clinical data sources, using a controlled vocabulary service and appropriate standards-based technological solutions. An inherent problem in the design of clinical trials is preserving the audit trail of the process through which these study eligibility criteria are constructed.

The key provenance requirement in TRANSFoRm is to maintain a uniform audit trail across the different software tools and configurations, and have it comply to Good Clinical Practice guidelines (GCP). This reflects the project's ambition to provide infrastructure for a wide variety of software, with shared key components: security, privacy, vocabulary services, and provenance. The designs of all software tools in TRANSFoRm are based on the Clinical Research Information Model (CRIM) [37] that specifies all process flows and data models that are required in TRANSFoRm use cases, and is GCP-compliant. Thus, CRIM also defines the level of detail required in the provenance traces.

3.1.1. TRANSFoRm provenance model

CRIM is provided as an Unified Medical Language System (UMLS) model, and as such unsuitable to be directly used for provenance representation. Firstly, its syntax is not expressed in terms of OPM or PROV and therefore not all their concepts fit the restrictions and logics behind provenance. Secondly, UMLS does not support semantic reasoning required for causality and temporal queries, which are both highly relevant to provenance. Therefore, instead of using it directly, CRIM was used as a basis for constructing the Randomized Clinical Trials Ontology (RCTO), which placed all concepts and actors from CRIM into an ontological structure, with explicit representation of processes which were implicit in CRIM. As the next step, Randomized Clinical Trials Provenance Ontology (RCTPO) was defined as extension of RCTO with OPM provenance concepts, such as versioning relationships between artifacts and create/edit actions, annotations of documents with their physical location, and links between agents, security information, and actions performed.

As an example, the concept of an electronic case report form is defined in CRIM as *eCaseReportForm*. In RCTO, it becomes an ontological concept, linked to a *ClinicalStudy* concept, which is, in turn, linked to a *TrialProtocolDocument* and an *AdverseEffectProcedureDocument*. In RCTPO, *eCaseReportForm* is an instance of an OPM artifact, and represents the form at a certain time point, with *laterGenerationThan* relationships to previous and future versions, *used* relations to edit tasks using it, *wasGeneratedBy* relations to the create/edit process that produced it, and all the structural links to domain concepts inherited from RCTO. Since CRIM, and consequently RCTO and RCTPO, implement Good Clinical Practice guidelines, this enables provenance records to be used in providing parts of the audit trail required by GCP. Full detail on how individual parts of CRIM were mapped to provenance concepts can be found in [38].

RCTPO contains some non-domain content as well in order to support the TRANSFoRm infrastructure middleware, message-passing and security. All the tools or applications used in TRANSFoRm are registered in the provenance store database, with the versioning and the link to the source/installation repository also maintained. Each time a user logs into an application his corresponding agent is retrieved or generated, a *userSessionArtifact* and a *createdSessionProcess* are generated and connected to authentication and authorisation processes. Following that, all consequent actions by the user in

the application are all associated to the original*createdSessionProcess*. This results in large interconnected graphs containing full audit trail of the processes, expressed in concepts that can be traced back to the CRIM conceptual model.

3.1.2. Provenance templates

As discussed above, the lack of a common execution environment complicates the capture of prospective provenance. Technologies such as D-PROV [39] support such provenance by relying on workflows as means of producing provenance data. TRANSFoRM did not have that option available, so the solution was to introduce *provenance template* as a higher-level abstraction of the provenance graph data. This construct specifies basic conceptual units that a software tool may record in the provenance repository, e.g. an edit operation on an eligibility criterion of a study, which is derived from the model and that is translated into a concrete provenance trace by a mapping model inside the provenance framework.

Provenance templates in TRANSFoRm use a graph syntax similar to OPM graphs. The difference is that their artifact, process, and agent nodes refer not to concrete instances in the past but to RCTPO domain concepts that shall be used for instantiation. Further graphical constructs are introduced to model subgraph pattern repetitions, node information description and physical or logical distribution of graph segments. An example of a provenance template is shown in Figure 2 describing a sequence of potential edits happening on eligibility criteria that are associated with a protocol of a clinical study. The multiple edits of the study, protocol and eligibility criteria are sequential repetitions, while different user sessions can be created in parallel by multiple users and software tool instances. A provenance graph segment created using the template is shown in Figure 3 depicting a scenario in which a researcher, TA, created a study with the protocol and eligibility criteria specifying that study participants should be over 60 years and female, while the other researcher, JR, modified it with another protocol specifying that participants should have Body Mass Index (BMI) over 40.

Associated with each entity is a set of annotations, so for example, the annotations for the eligibility criteria version 57_17 are:

```
Details for ClinicalStudyEligibilityCriteria 57_17:
WasDerivedFrom: OPMArtifact_50
WasGeneratedBy: OPMProcess_55
```

```
AnnotationTime: 2013-10-03 07:01:56
Type: ClinicalStudyEligibilityCriteria; Artifact
Id:  ItInst_1150449877_1380780116960_objectA2_
rcto:ClinicalStudyEligibilityCriteria
Value: Inclusion Criteria: Age > 60; Inclusion Criteria: Gender = F
```

The provenance data captured in this way are stored in a semantically annotated database for later auditing and analysis. All provenance entities contain at least one annotation: their domain semantic type defined in RCTPO and derived from the Clinical Research Information Model (CRIM). The time of annotation, the TRANSFoRm module which generated the data, and the physical allocation of the data itself are also always available. The database is implemented in a standard relational DBMS, but supporting both SQL queries, and SPARQL queries that can make use of semantic annotations. The latter is achieved through the Data-to-Relational-Query (D2RQ) tool [40], which provides mappings between relational schemata and OWL/RDFS ontologies.

3.1.3. Securing provenance in TRANSFoRm

Within the TRANSFoRm architecture, the security solution layer is in charge of ensuring the identities of TRANSFoRm users, and managing their allowed actions. The security layer performs this by issuing SAML [41] assertions to authenticated clients to establish that they are allowed to perform a certain action. The provenance record of those actions is maintained separately from the security layer and it contains the SAML assertion itself, which holds no identifiable information, but allows invoking the security layer's API to extract additional information, e.g. why was the assertion issued or how was the requesting user authenticated. The API calls are managed by the security layer, which can decide whether further information is available about the SAML assertion for the user posing the question.

3.1.4. Queries and result visualisation

For auditing and querying of stored provenance data, TRANSFoRm provides a lightweight web-based query tool that supports pre-designed SQL and SPARQL queries, addition of new queries, and visual exploration of the provenance items by browsing the graph structure. A initial set of useful queries enables users to browse the provenance graph and retrieve information about clinical trial studies. Authorised users can create, execute and

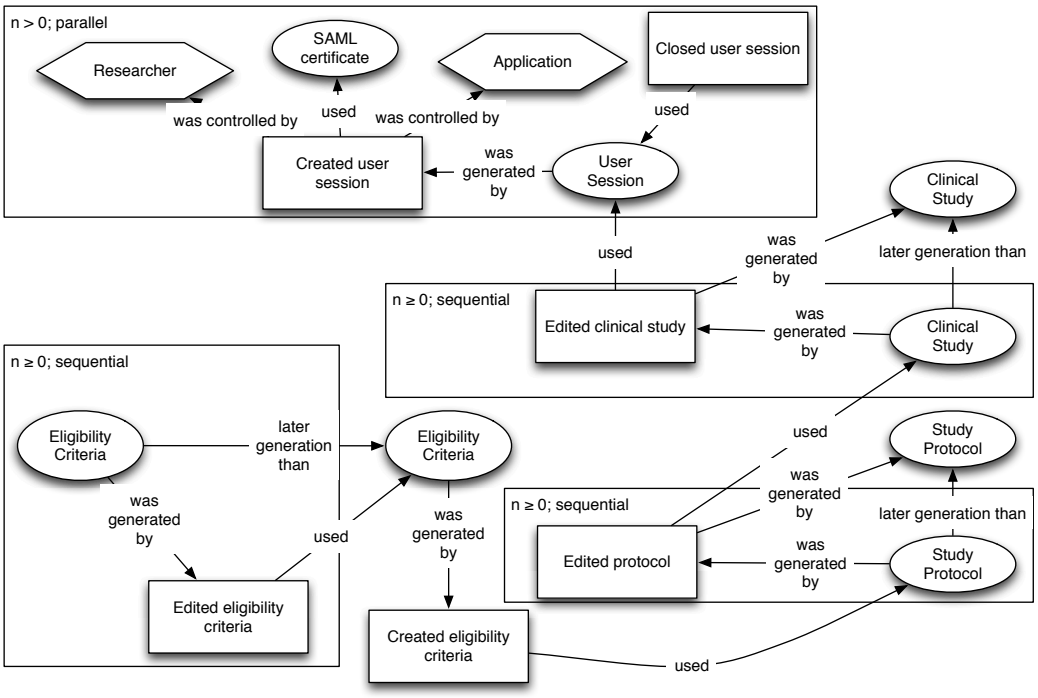

Figure 2: A TRANSFoRm provenance template taken from Query Formulation Tool, describing creation and editing of eligibility criteria, as specified in the CRIM model

share new queries through the tool. The tool consists of two parts: Query Collection and a Provenance Browser, as shown in Figure 4. Predefined queries, shown in the sidebar on the left, answer specific questions from the provenance data, with the results displayed as tabular or chart outputs. The queries are parameterised with ontological concepts taken from the model, thus allowing close mapping to the healthcare domain.

The provenance entities in the tabular results from the queries contain hyperlinks that connect to the Provenance Browser tool that provides interactive graphical navigation through stored provenance graphs. The user clicks on nodes to reveal further detail about the process through ontological annotations on each node, thereby gaining access to the wider provenance information surrounding the item of interest. The queries available range from security-focused to ones detailing the clinical study processes. The following are three query examples in the tool:

1. List all processes together with their authentication certificates.

```
select ?p, ?said
where{
    ?p rdf:type opmo:Process .
    ?p urn:oasis:names:tc:SAML:2.0:assertion#ID ?said .
}
union
select ?p, ?said
where{
    ?p rdf:type opmo:Process .
    ?p (opmo:cause/opmo:effectInverse)* ?entity .
    ?entity urn:oasis:names:tc:SAML:2.0:assertion#ID ?said .
```

2. List all the initiating processes of all the queries that were completed on 01/04/2012. The query uses SQL for efficiency.

```
SELECT OPMProcess.ProcessKey
FROM OPMProcess, OPMDependences
WHERE OPMProcess.OPMProcessEndTime BETWEEN 01/04/2012 AND 02/04/2012 AND
    NOT OPMDependence.OPMDependenceEffect = OPMProcess.OPMProcessKey
```

3. Which databases were used to retrieve patient information of the cases of a specific clinical trial study with artifact identified by clinicalStudyArtifactURI?

```
select ?database
where{
    clinicalStudyArtifactURI rctpo:hasEligibilityCriteria ?eligCriteria .
    ?p rdf:type rctpo:EligibilityCriteriaQueryExecution .
    ?p opmo:Used ?eligCriteria .
    ?p1 rdf:type rctpo:EHRDBQueryExecution .
    ?p1 opmo:wasTriggeredBy* ?p .
    ?p1 opmo:Used ?database .
    ?database rdf:type rctpo:eHRDatabase .
}
```

3.2. EHR4CR (Electronic Health Records For Clinical Research)

The EHR4CR project aims to develop an integrated reusable solution to seamlessly connect existing clinical research platforms and healthcare networks across multiple European countries and legal frameworks. An EHR4CR service and data integration platform will form an interoperable

and scalable infrastructure that connects the patient data information systems in hospitals. The platform supports protocol feasibility study management and automated queries that allow a clinical researcher to dynamically discover eligible patients within the hospitals and recruit them for conducting clinical trials. The automated patient identification and recruitment, and clinical trial management capacities will help to speed up the protocol feasibility study and patient recruitment processes for a clinical trial and reduce the costs. The platform will later support clinical trial execution and adverse event reporting.

The main functionality of the EHR4CR platform is acquisition and integration of data from distributed heterogeneous healthcare information resources. Queries for eligible patients are issued by clinical researchers via a web-based workbench, the queries are distributed to endpoints (data sources at distributed hospitals) by a query orchestrator. Each endpoint accesses a data warehouse, which is populated with data from electronic healthcare record (EHR) systems at each hospital using an extract-transform-load (ETL) process. The results from each endpoint are aggregated and returned to the researcher. The platform requires many supporting services to ensure security, translation of terminology between the different medical models used at different sources, and a registry of endpoints, among others. Given the distributed nature of the system and its inherent regulatory and governance issues, replicas of each type of service are deployed in several European states. These replicas are interconnected for enabling service integration and data exchange with sufficient access control. This unavoidably increases the complexity and difficulty for maintaining traceability and auditability of system transactions. Performance, data quality, privacy protection and regulatory compliance are among the major non-functional requirements.

Provenance data in EHR4CR is captured during the execution of queries, and in the supporting data handling and access control processes. It provides reasoning information for answering questions that concerns data access, billing, study management, and data quality issues. Motivating provenance questions in EHR4CR include:

1. How did a feasibility study evolve into its current version?
2. How was a query result produced by the system?
3. How was an authorisation decision made in a system?

EHR4CR takes a model-driven data transformation approach for populating standard hospital patient data into a data warehouse to prepare infor-

mation in a common form across hospitals, to allow queries to be expressed, distributed and answered by the multiple sources. Records of the ETL processes are critical for ensuring whether, and to what extent, the data can be relied on.

The EHR4CR platform supports dynamic hospital and patient discovery with a number of hospitals connected to the platform. Provenance graph data is attached to each query result to explain how the result was generated. The provenance mechanism records which data sources were involved in answering a query, and provenance information related to resource discovery and binding processes to help the clinical research understand the query results and improve the study design strategy, as it indicates what range of sources are available or are frequently offline. The provenance data can also be used for addressing billing issues involved in the clinical trails.

Privacy protection and security compliance are addressed systematically in the design and implementation of EHR4CR. Data access, authentication and authorisation processes are all recorded, and it is intended that structured provenance data will replace audit logs, and allowing the interlinking of locally generated audit trails to facilitate improved auditability for local and cross-platform auditing. Sensitive information is not contained in the provenance data returned to the researcher, but is stored locally in each subsystem in different management realms to allow auditing and further investigation into specific issues with finer granularity.

Finally, a clinical trial study changes over the course of time with refinements to eligibility criteria and repeated queries to the system. The provenance of a study can provide a visualisable history of the study and the reasoning behind it. This requires integrating the provenance of multiple query executions with a record of how the study itself, at the clinical researcher's site, has evolved over time.

As is the case in TRANSFoRm, EHR4CR records provenance according to templates, which specify the form of provenance to be captured in an instance-agnostic form. This enables provenance queries to be expressed, as the structure of provenance data is known at design time. EHR4CR uses the W3C PROV data model rather than OPM in TRANSFoRm, and does not make use of bridging ontologies or a common information model such as CRIM. Figure 5 shows a provenance template for documenting the execution of a query in a component of the platform. The security-related processes and data are indicated in grey, and form a subgraph for queries regarding security and access control.

In EHR4CR, provenance data is stored locally to each component in an RDBMS. It is not stored in the form of PROV statements, but can be exported from a component as PROV, which is used as the transport mechanism. In particular, a non-sensitive subset of provenance data is returned from the hospitals used as data sources to the clinical researcher embedded in the XML message containing the eligibility query results. This data can then be incorporated with the researcher's own provenance graph, regarding the local refinement and execution of their study queries. The result is that the researcher can query the provenance of their eligibility query results in a local store.

We present below two brief examples of where provenance is currently implemented and used in EHR4CR, particularly to answer question 2 above "How was a query result produced by the system?". These will be referred to in the following section for illustration.

3.2.1. EHR4CR Extract-Transform-Load (ETL) use case

Each hospital EHR database is different, with a different schema, and with different quality and types of data present, and one hospital can have multiple EHR databases. EHR4CR requires a common data warehouse at each site, to allow distributed querying. The overall template for recording the provenance of these actions is shown in Figure 5. Because of the heterogeneity, in the ETL process that transforms data from the EHR databases to the data warehouse, different assumptions will be made at different hospitals. A key purpose of provenance is to record the assumptions that have been made in the data transformation. For example, to preserve confidentiality, dates of birth will be set to the first day of the month or year to inhibit identification of the patient. Where data is repeated across EHR databases, the one from which a fact is drawn must be recorded. There may be several dates associated with a specific lab test such as when the sample was taken, when it was received by the lab and when the analysis was performed, and the warehouse requires a single date and therefore one of these must be chosen. Observing the provenance of an ETL process allows users to understand what data can be relied on and to what detail.

3.2.2. EHR4CR filtering use case

There can be no assumption that the ETL process or the original data in the EHR systems is flawless. Therefore during the determination of eligible patient counts, certain clinical facts in the warehouse may have to be

discarded because, for example, an event date was missing or a unit of measurement was not in a standard representation. Thus a set of filters are used to exclude incomplete or defective facts. A record of this filtering process is part of the provenance returned to the clinical researcher. The discarded facts cannot themselves be revealed to the researcher but counts of facts accepted or rejected by each filter may, so the provenance is annotated with these counts. The counts are used by the researcher to gauge the quality of the data and where the discard rate is very high, may explain why the patient counts are low or zero. Thus, a provenance query on finding filter instances which rejected 4 patients returns an RDF:

```
_1 rdf:type DobFilter.
_2 rdf:type Missing.
_2 IsOutcomeOf _1.
_3 rdf:type Counter.
_3 IsCounterOf _2.
_3 rdf:value 4.
```

The result is returned as metadata inside the protocol feasibility query result, and then displayed in a simplified form in the user interface of the clinical researcher.

4. Recommendations

Based on our experience in the TRANSFoRm, EHR4CR and earlier projects, we now present a set of recommendations concerning how to model the provenance data, and the mechanisms for its capture, storage, security, and querying. These are described at a high level, to ensure they are reusable across systems yet to be designed, but with rationale provided so that developers can understand how they should apply to their own systems.

4.1. Modelling

A key aspect of healthcare software systems is the distribution of authority and control. At the time such a system is first deployed, the set of sources and consumers of healthcare data will not be fixed, and those that are initially included will have their own local policies and practices. To be able to answer provenance questions in such a system, all parties need to record *interoperable* provenance information to a common specification. The modelling allowed by that specification should also be rich enough to describe the

processing, exchange, ownership, temporal aspects etc. of the data involved in a computational way, else answering the questions over such large datasets will be infeasible.

The de-facto standard for rich, interoperable provenance data is, currently, the Open Provenance Model (OPM). This has been used in a wide range of systems and there are a number of publicly available libraries and tools for it [42]. The forthcoming W3C PROV data model (PROV-DM) and accompanying specifications provide the first official standard. While currently in the final stage of the standardisation process at time of writing, it is expected to become a standard during 2013. W3C maintains a registry of PROV implementations [43]. We note that PROV is strongly influenced by OPM, and differs primarily in adding flexibility for describing the provenance of mutable resources and for conveniently ascribing attribution, both aspects relevant to healthcare data.

Both OPM and PROV contain the basic building blocks needed for provenance-aware applications, however they should not be considered the final word on the matter. There is frequently a need to represent structural domain information behind provenance construction, and to include it in the provenance traces. D-PROV, mentioned above, does that for workflow style applications, capturing workflow graph structure, input/output data ports and other relevant entities.

Recommendation 1. (Syntax) *Use W3C PROV or the Open Provenance Model for modelling provenance data recorded in an electronic healthcare system. When choosing between the two, consider available tools and libraries in both systems for potential reuse or adaptation.*

Example: TRANSFoRm uses the RCTPO extension of OPM, containing domain concepts and relationships.

Example: The EHR4CR ETL and filtering use cases both use an extension of PROV.

Generic provenance models, such as OPM and PROV, must be extended with a domain- or application-specific vocabulary in order for provenance to be informative. As with any vocabulary, usage of terminologies proprietary to individual institutions greatly hampers interoperability. For the purposes of informative querying, the querier must know something about the terms used in the provenance graph.

Recommendation 2. (Vocabulary) *Institutions involved in a distributed healthcare application should agree to a common vocabulary to use in provenance constructs where possible.*

Example: TRANSFoRm uses a generic interoperability data model [44] combined with a terminology server [45] to ensure consistency across data sources.

Example: The provenance templates used in EHR4CR filtering and ETL processes employ a vocabulary, which is then common across the hospital sites which locally record the provenance.

In addition to vocabularies, there is a large amount of well-standardised information and process models to describe the domain data (e.g. EHRs, clinical trials, drugs). Data provenance concepts associated to concepts in standardised domain models should maintain the semantic relations between them. Linking domain and provenance-aware models allows for a better description of procedures and dataflows, the maintenance of a bridge between both domain and provenance data and, consequently, the possibility of performing richer queries. Models should be shared with the community in order to facilitate data interchange and the improvement of one's own data modelling process. Provenance data have to describe the state of a system in a particular moment, and domain data evolve in time, thus provenance storage modules have to provide a mechanism for maintaining the minimal useful information that reflects the data state, and provenance access policies have to address the issue of restricting domain data access during query executions.

Recommendation 3. (Domain models) *Link provenance models and data with domain knowledge models and data, respectively. These may be expressed in the form of templates, similar in form to workflows or business processes, and should be shared with the community.*

Example: TRANSFoRm RCTPO and RCTO ontologies are publicly available and other randomized clinical trials can use them to annotate their data, compare results and methodologies and choose the most suitable procedures accordingly.

Note that we do not prescribe particular ontologies here. There is a wide array of sometimes overlapping biomedicine-related ontologies, and the choice of which to use in a given application is separate from the decisions on provenance infrastructure. Aside from ontologies of biomedical concepts,

there will be concepts that may be captured in the provenance, whether as typing information on the provenance graphs, or additional context information. In some cases, these may be common across multiple distributed biomedical applications, e.g. due to the varying quality of stored medical data, many applications will require filtering of records before use by remote services, and describing this in the provenance requires some vocabulary. Defining common vocabularies for such cross-application provenance is certainly beneficial, but beyond the scope of this (or any single) paper.

When considering how to model provenance, it is natural to ask both "Are we recording the provenance in enough detail to answer the questions that will be asked?" and "Are we reducing performance and increasing storage requirements by recording too much provenance data?" One observation is that it is rare that all possible provenance questions can be known in advance: when provenance data is present, users will realise new questions they wish to ask. This leads to two related recommendations.

Recommendation 4. (Detail focus) *Always aim to model the detail of what happens, including each processing step and data item involved, rather than summary information that directly answers current provenance questions.*

Example: In the EHR4CR filtering use case, we do not simply record a count of how many records have been filtered on which grounds, but express the filtering process, with inputs and outputs to/from each filter. This allows the filtering provenance to be interlinked with the provenance of the query as a whole.

Instead of simply recording that "the figures produced in this feasibility study are drawn from these hospital data sources", because this is the primary provenance question being answered, we recommend modelling the steps that lead from the hospital data source to the creation of the feasibility study. Then, if later questions are asked about the influence of the infrastructure on the feasibility report contents are asked, e.g. "Were there timeouts that meant a particular data source yielded a possibly incorrect value of zero patients?", then this can be answered with the given provenance information. As there is no way to return to the past, the question could not otherwise be answered.

While the above recommendation helps to future-proof the provenance somewhat, it does not answer the question of the *level* of detail to record. If every step of every process is recorded to the lowest level that is technically feasible, the performance and storage overhead may be unacceptable.

We need to determine at what *granularity* the provenance should model processes. One influence is that the medical domain includes many existing semantically rich models and ontologies, and the contents of the provenance and the granularity at which it is modelled should be dictated by these as far as is feasible. From our experience, we suggest the following additional guideline. For a methodological approach incorporating these two related principles in provenance recording, see our existing work in this area [46].

Recommendation 5. (Granularity) *Use the existing biomedical models and ontologies as indicators of the level of granularity of process description that users are likely to be interested in the future, and validate it against the provenance questions known at design time.*

Example: TRANSFoRm provenance graphs are defined in terms of templates that are designed in collaboration with the software developers who use them to commit provenance graph fragments, and users who will need to analyse provenance audit trails.

Example: The EHR4CR ETL provenance works at the level of a set of clinical facts of the same type and from the same source. Thus, all diagnoses or all results of a specific lab test will have a single provenance record if they hailed from the same source. For a typical data warehouse, this will run into tens or hundreds of provenance records. If the provenance were applied to each individual fact separately, the provenance data would likely expand, by an order of magnitude, the original data warehouse, which may already have over 10^7 rows.

A danger, when considering how to model provenance, is that the perspective of one user group will suggest that provenance be separated into isolated records in a way that makes it difficult to answer queries of other groups. For example, from the perspective of a clinical researcher, it may be natural to consider each study having its own provenance, with no or limited connection between these provenance records. This view may be encouraged by the privacy and intellectual property concerns that the author of one study should not have access to the provenance regarding another author's study. However, considered from a data provider's perspective, or that of the organisation providing the distributed infrastructure, the division between studies is not a relevant distinction. The same data item may be accessed within multiple studies and for the data provider's audit purposes, that is useful provenance knowledge. OPM and PROV allow, but do not

require, provenance to be modelled as a single graph, in which all parts are interconnected, because this is an accurate representation of what has occurred, e.g. multiple studies can use the same data source, one study can use multiple data sources, and they all use a common distributed infrastructure. Note that the same thing may be identified in different ways in the provenance recorded by different organisations/actors so linkage mechanisms should be present.

Recommendation 6. (Connectivity) *Model the provenance with the expectation that, if brought together, it would form a single graph describing the full, interconnected history of the system, as opposed to being delimited into a set of isolated records. Globally unique IDs should be used where feasible to facilitate interlinking.*

Example: In TRANSFoRm, each entity in the stored graph has a globally unique identifier, which enables connection of graphs created by different software tools, for example Authentication Framework and Query Workbench described above and shown as two colours in Figure 3.

Example: In EHR4CR, each query executed has a globally unique ID, as does each hospital. Therefore, in the EHR4CR filtering use case each filtering step in the filtering use case can be given a unique ID simply by enumerating the steps and combining with the query ID, and each count produced by a filter likewise.

Provenance describes not only automated processes but also human actions relevant to the data and models in the system. Human interactions with the electronic system, and human decision-making informing those interactions, are important factors in how the system operates. For example, the approval for patient data to be used as part of a clinical study may be affected by decisions of doctors, the patient and potentially national bodies and others. Understanding the provenance of particular study results requires knowing, and so having captured, the key human actions and decisions.

Recommendation 7. (Human actions) *Include both salient human activities and automated processes in the model of provenance to be captured.*

Example: TRANSFoRm provenance model, RCTPO is derived from CRIM, an information model comprising human workflows as well as technical steps, and thus contains concepts modelling human decisions and actions.

Example: In the EHR4CR platform, the selection of patients can only be partially automated and can at best produce a shortlist that would be narrowed down using clinical judgement. These clinical decisions will have to be recorded (the project implementation has not reached this stage at time of writing).

More generally, we can distinguish three key aspects that should be considered in an application's provenance model. First, the process flow indicates the order in which events occurred and where an action was responsible for indirectly causing a consequence elsewhere in the system. Second, the data flow describes where data originates and ends up, and sometimes also what value it has. Finally, the responsibility for actions that occur should be considered.

Recommendation 8. (Model elements) *In designing the provenance model, consider explicitly representing each element of the process flows and the data flows, as well as the attributions of actions to users. Layering ontologies is a useful approach that maintains logical separation between these distinct elements.*

Example: TRANSFoRm uses the RCTPO ontology to describe the process structures and relationships between actions and users, while OPM is employed as the basis for interactions between data and actions.

There are provenance-related mechanisms in many existing systems, including those of the individual institutions involved in federated healthcare systems. For example, documents will often be version controlled, auditing will occur for individual databases, software will write logs to aid later debugging, etc. Provenance aims to generalise over these different forms of recorded history, each of which may contain relevant information for local and remote users. The existing information could, in some cases, be included into the provenance data through translation, e.g. relevant entries from a database log could be parsed and translated into OPM or PROV data. Alternatively, it could be included by reference. For example, in TRANSFoRm, the provenance graphs contain links to logs of where authentication has occurred. The latter data has tighter security controls applied than the provenance that links to it.

Recommendation 9. (Reuse of existing data) *Where feasible, integrate the data captured by existing mechanisms (version control, audit, logging, etc.)*

into the provenance record, whether by reference or translation, to provide as rich and integrated an account as possible.

Example: TRANSFoRm provenance graphs do not store the authentication information beyond the user identifier. Authentication details are captured as a SAML [41] certificate which can be used for further audits, if needed, outside the provenance infrastructure.

4.2. Capture

The optimal approach to capturing provenance depends on the nature of the software system being observed. In the presence of a shared execution middleware (e.g. workflow engine, query processor), it is convenient to embed provenance support into the middleware, and let it record every action. If such middleware does not exist, but there is a shared process model between the tools, each tool can submit template-based graph fragments to a centralised service, that are compliant with the model definitions. Finally, in the absence of both the shared middleware and the shared process models, tools themselves can be left to define their own provenance traces, and trust that there will be a consistent entity naming schema to allow for cross-tool queries.

Even in the latter two cases, the provenance capture functionality does not necessarily have to be created from scratch. Libraries for modelling and serialising fragments of provenance within application code exist for both OPM and PROV [42, 43]. Currently, Java has the widest support in terms of libraries, but other languages, such as Python, have some implementations, and RDF and XML are most supported for representing the provenance data. The advantage of such libraries is that, along with the provenance models themselves, they have a community that use them and give feedback to improve them, which a proprietary implementation would not.

Recommendation 10. (Library reuse) *If not using shared execution middleware with provenance capture support, make use of existing OPM/PROV libraries to capture provenance data within your application code.*

Example: In order to minimise the code maintenance effort, future development of the EHR4CR and TRANSFoRm provenance infrastructures

intend to use the PROVoKing library[1], which was not available at the time the project software was designed.

Recommendation 4 above suggested that, because not all provenance questions are known in advance, it is preferable to record the key facts about what has occurred, events that occur, data used, and individual's responsibility, and not just the answers to the provenance questions known at design time. As we cannot return to the past, not documenting some event at the time it occurs can mean that certain future provenance questions can never be answered. This modelling decision then leads to an associated recommendation regarding capture.

Recommendation 11. (Timely capture) *Build the runtime provenance capture functionality into each step of the system processes, at the appropriate level of granularity, using global identifiers where possible and linking to records of preceding steps using the chosen provenance model relations.*

Example: TRANSFoRm software tools write template-based provenance graph fragments as they execute, at specified time-points.

Example: In both the EHR4CR filtering and ETL use cases, each step of the filtering/transformation process is captured as it is executed.

With regards to performance, provenance capture (as with logging) can have an immediate and pervasive impact on execution speed. For example, a prior study reported an overhead of just under 10% due to provenance capture in an application[47]. This can sometimes be mitigated by intelligent engineering, but should not be overlooked.

Recommendation 12. (Performance testing) *Test the performance overhead of provenance capture within a small sample part of your application to ensure it is acceptable for your application or if the technologies used and level of detail of model need to be modified.*

Example: During development, each TRANSFoRm tool performed performance tests before and after introducing provenance support, and in some cases the frequency of provenance capture was reduced so as not to adversely affect the performance of the tool. Similarly, the usage of provenance templates facilitates profiling various stress levels on the provenance server by simulating the load under different usage patterns and frequencies.

[1] https://sites.google.com/site/provokinglibrary/

4.3. Storage

Provenance graphs are the most popular way of representing provenance information, used in both OPM and PROV, and they can consist of thousands, even millions of nodes with associated attributes. As such, they represent a special case of graph storage management allowing the use of same basic indexing techniques, e.g. clustering and block partitioning, which is provided by NoSQL databases such as Neo4J [48]. However, graph data can also be stored in classical relational databases as well. The D2RQ language [40] describes mappings between relational database schemata and OWL/RDFS ontologies, allowing access to RDF views on a relational data. Data partitioned in such a way can often be conveniently distributed among several sites, which may be necessary, depending on the institution's data regulations. Furthermore, in such scenario, a central query resolution service is then needed to determine if some provenance query will run across multiple sites, and if so to resolve the node identifiers involved.

Recommendation 13. (Database structure) *Consider whether provenance storage needs to be provided in a relational database, if one is the standard solution in the research environment, or would an RDF store or a NoSQL graph database be an option. Privacy and data regulations will play an important role in deciding whether and how best to distribute the data in a multiple site scenario.*

Example: TRANSFoRm data uses several ontologies and often needs to be queried based on terms from those ontologies, so a D2RQ layer is used over a relational database to support such queries.

Example: The EHR4CR ETL provenance data is currently stored within a relational data warehouse since it affords querying the clinical data and its provenance data by the same mechanism, i.e. SQL and querying is less dependent on ontological terms.

Provenance capture is data storage of the same kind as any other data storage in the application. For example, eligibility crtieria query results will be received and stored at a researcher's site, while a copy of the query will be cached at the hospital for auditing. It makes sense, therefore to reuse application functionality for provenance capture. In some cases, it makes sense to store provenance data centrally, e.g. the storage and maintenance of provenance information may be considered a matter for middleware rather than clients. This means events can be documented remotely to where they occur.

Recommendation 14. (Data infrastructure) *Reuse existing application infrastructure for provenance transmission if storing provenance centrally rather than at each site. Messaging systems may be used to transmit data reliably and asynchronously between application sites.*

Example: TRANSFoRm provenance data is stored centrally, and transmitted via the existing middleware web service infrastructure.

Example: In the EHR4CR filtering provenance use case, the provenance captured by the hospital is encoded as RDF and is attached to the XML message containing the query results that is returned to the clinical researcher. No additional transmission infrastructure needed to be implemented.

Provenance data will grow over time at the rate that depends on the patterns of graph segments that applications use [49]. For example, re-running an existing ETL process would create fewer new nodes than creation of a new clinical trial. Therefore, it is important to profile the frequency of various provenance-generating actions. With regards to optimising storage use, deletion policies may need to be devised to periodically remove data deemed unnecessary for meeting regulations. To ensure that the remaining data is still usable, it is advisable to maintain interlinked provenance graphs when deleting data. For example, even if deleting much of the detail about how an old EHR4CR feasibility study was executed, such as to which hospitals the query was sent and what statistics were returned, it may be valuable to keep the link between the query and its results in the provenance graph, as this connection is used in historic queries, e.g. looking into rates of query success.

Recommendation 15. (Rate of growth) *Due to the typically large size of provenance data collected over time, it is crucial to establish early on the rate at which provenance storage requirements are expected to increase over time. The level of detail in the model may need to be adjusted accordingly and some deletion/archiving procedures introduced.*

Example: The usage of provenance templates in TRANSFoRm enables reliable estimates of growth based on the application usage profile, since the size of provenance data committed in each template is known.

4.4. Security

A new set of security considerations arises for provenance data in relation to regulating access to records of a resource, rather than the resource itself,

e.g. whether a user is allowed to access provenance trail of a certain process, or of a process affecting a certain data item. Even if a data item itself is not accessible to the user, this should not necessarily restrict the user from accessing some information about it – an auditor may not be allowed to see a patient's full electronic health record, but may see that the patient was entered into a clinical trial. Therefore, the policy for access to the provenance data items, while influenced by the access policy to subject data items, is not necessarily identical to it.

Recommendation 16. (Permissions) *When implementing provenance access control, consider the differences in access permissions between concrete data, and provenance records of that data.*

Example: As part of its security system design, TRANSFoRm has produced an Operational Security Policy document, containing an access control model, which explicitly addresses both the various types of data and the provenance records of those data.

Since provenance data are typically stored in relational databases or RDF stores, one approach to access control is to leave the access control to the storage security mechanisms. Security for database technologies has been studied extensively in the past (see [50] for an overview) and the increasing need for maintaining semantic resources has developed a large set of works focused on security for RDF data [51, 52, 53].

Specific policies for provenance data can be implemented either by using either a dedicated provenance access control language [54, 55, 56], or an existing standard such as OASIS eXtensible Access Control Markup Language (XACML). Since XACML is not generally suitable for RDF data [57, 54, 58, 59, 60], extensions to it are necessary [61, 60].

A key feature of a provenance access control language is a mechanism to obfuscate the parts of the graph which the user is not allowed to view, but which are deemed safe with the details omitted [62]. This can be done by, for example, introducing abstract entities and processes, and by anonymising certain nodes. Such a mechanism needs to transform the provenance graph based on the access privileges of the user posing a query, hiding or abstracting certain parts, before assembling the query response from the transformed graph. Similarly, the ProPub system [63] computes views over provenance graphs for the purpose of publication by satisfying a set of privacy requirements, specified by the user in terms of anonymising, abstracting, and hiding certain graph segments.

Recommendation 17. (Restrictions) *When designing an access control mechanism for provenance data, decide whether restricted provenance information should be completely hidden or just have details abstracted. If the latter is the case, a mechanism is needed for answering user queries using graphs restricted to that user's access level.*

Example: TRANSFoRm has designed a method for abstracting portions of the provenance graph when answering queries so as to maximise the amount of information required, while observing the security constraints [64].

Example: By using counts, the EHR4CR filtering provenance data provides useful information about data quality whilst not compromising the confidentiality of patient data.

By its nature, provenance data may act as an additional layer of security control, that is semantically enabled and thus more closely linked to the application's domain logic. For example, an act of extracting data from a data source may need to have record of the authorisation made to allow that access. However, duplication of sensitive data, such as usernames and passwords, is rarely desirable or, indeed, feasible, and so a solution needs to be found to link security data with provenance records.

Recommendation 18. (Sensitive data) *When provenance is used to capture data about the actions of the security layers, separation of functionality needs to be introduced to avoid inadvertently exposing sensitive data. One way of doing so is by providing token identifiers which can then be used to request detail from the security mechanism.*

Example: In TRANSFoRm, each application can use the provenance API to annotate the data provenance details and link them to its domain data in an encrypted format. The access to this encrypted domain data, e.g. user authentication details, has to be approved by the relevant module, typically the security layer or the application that has performed the provenance capture.

4.5. Querying and visualisation

Provenance data is potentially accrued through every use of tools in a software system, and the scale of collected data introduces a degree of complexity when designing user-facing analytics. The analysis and reporting on provenance data can be done by simply executing predefined queries based

on some pattern and presenting them graphically, e.g. what are the properties of a query used to recruit patients for a particular clinical study, or how was a particular process instance authenticated.

An important issue when designing the provenance data queries is to allow for the predefined queries to evolve and be extended with new queries. Introducing user mechanisms for adding queries to the system ensures that new questions can be asked from the data as the understanding of the system increases.

Recommendation 19. (Extensibility of queries) *Introduce mechanisms for dynamically adding new provenance queries to the software.*

Example: TRANSFoRm provenance query tool provides several categories of parameterised queries, defined in SPARQL. These queries are accessible in an XML configuration file on the server, where a system administrator can simply add new items as needed.

Further value in querying is provided by allowing the user to examine provenance graphs by posing open-ended queries and following the trail of the processes and data instances to find out a greater detail of the process. This mode of interactive, iterative querying can be supported using visualisations based on the graphical representation of provenance data. Granularity of the queries needs to be considered as well: a common pattern is to start with a high-level query and then gradually narrow the query down, which should be reflected in the query tool design. Care needs to be taken when designing the user interface not to overwhelm the user with the scale of the conceptual graph they are traversing.

Recommendation 20. (Interactive querying) *Provenance data can be used to answer individual queries or create tabular and graphical reports, but added value can be obtained by exploiting the graph representation of provenance to support open-ended, investigative querying.*

Example: TRANSFoRm provides a query interface which supports a graph view of the provenance entities in the response, allowing the user to browse their surrounding and gain deeper understanding by directly examining the entity annotations.

If the underlying data store, relational or non-relational, has explicit support for OPM nodes and edges, it is possible to separate queries that depend

on domain's semantic annotations, e.g. concepts from a clinical terminology of some hospital in EHR4CR, and those that are purely expressed in terms of provenance concepts (all items using a certain data set). The latter can often be optimised by breaking the abstraction and encoding them directly in the raw query language of the database used.

Recommendation 21. (Breaking abstraction) *When creating provenance queries, whether for atomic questions, or for navigating the provenance graph space, try to find queries that can be optimised by being directly formulated in the underlying query language and investigate whether the performance gain warrants this.*

Example: TRANSFoRm uses D2RQ to provide a SPARQL interface to a relational data store where the provenance data resides. While most queries are posed in SPARQL, there are some frequent operations that are directly implemented as SQL for efficiency purposes. Typical example are the TRANSFoRm live activity reporting portal components which may be in constant use by multiple users.

Example: In the EHR4CR ETL case, the provenance is abstracted and kept in the data warehouse's SQL database. Instead of putting the provenance graph itself in the database, there is a well-defined mapping from the columns in the database to a provenance graph (following a template).

5. Conclusions

The software landscape in biomedical research is vast and disconnected, encompassing electronic health record systems, genetic data repositories, clinical trial systems, and diagnostic support systems, among others. Research tasks are executed by numerous actors, including researchers and clinical staff, from multiple organisations and with different data governance restrictions. In order to ensure that the research results are auditable, verifiable, and reproducible, the provenance of both data and processes involved needs to be captured in a consistent, interoperable, confidentiality-preserving manner across all software tools used in the research task.

While progress has been made in developing standardised, extensible, flexible provenance models suited to represent this provenance data, this alone does not mean that the way to model, capture, securely store and query provenance in a given application is a trivial problem. In addition, there

are multiple levels of modelling involved, starting from low-level provenance specific constructs, to higher level ones that correspond to domain and application concepts that are often taken from existing models. Developers need not only to know which technologies to use, but what principles to follow in their designs.

In this paper, we have laid out a set of recommendations for implementing provenance through analysing our experience in developing two large-scale biomedical research systems. These are not meant to be definite and immutable, but rather a starting point for researchers that will eventually be enriched and improved by contributions of others. Ultimately, as the provenance technology matures, this work will give rise to more formal software engineering techniques that will facilitate provenance implementation across a broad range of software tools in biomedical domain and beyond.

6. Acknowledgements

The authors would like to thank Prof. Theodoros Arvanitis, James Rossiter, and the rest of the team at University of Birmingham who designed and implemented the TRANSFoRm Query Workbench. The provenance data shown was created during their user tests.

We would also like to thank the reviewers of an earlier version of this paper for helpful comments and constructive suggestions.

References

[1] L. Moreau, The Foundations for Provenance on the Web, Foundations and Trends in Web Science 2 (2010) 99–241.

[2] D. Hull, K. Wolstencroft, R. Stevens, C. Goble, M. R. Pocock, P. Li, T. Oinn, Taverna: A tool for building and running workflows of services, Nucleic Acids Research 34 (2006) 729–732.

[3] European Medicines Agency, ICH Topic E6(R1) Guideline for Good Clinical Practice, Technical Report, European Medicines Agency, 2002. URL: `http://www.ema.europa.eu/docs/en_GB/document_library/Scientific_guideline/2009/09/WC500002874.pdf`.

[4] K. F. Schulz, D. G. Altman, D. Moher, CONSORT 2010 Statement: updated guidelines for reporting parallel group randomised trials, Bmj 340 (2010) c332–c332.

[5] E. von Elm, D. G. Altman, M. Egger, S. J. Pocock, P. C. Gø tzsche, J. P. Vandenbroucke, Strengthening the reporting of observational studies in epidemiology (STROBE) statement: guidelines for reporting observational studies, BMJ 335 (2007) 806–808.

[6] Clinical Data Interchange Standards Consortium, CDISC Analysis DAta Model v1.2, 2012.

[7] Space Time Research (Institution/Organization), The Open Data Initiative, 2011. URL: http://www.opendatainitiative.org/home/about.

[8] T. Groves, F. Godlee, Open science and reproducible research, Bmj 344 (2012) e4383–e4383.

[9] S. Miles, P. Groth, M. Branco, L. Moreau, The requirements of recording and using provenance in e-Science experiments, Journal of Grid Computing 5 (2007) 1–25.

[10] P. Groth, Y. Gil, J. Cheney, S. Miles, Requirements for Provenance on the Web, International Journal of Digital Curation 7 (2012) 39–56.

[11] P. Buneman, S. Khanna, W. C. Tan, Why and Where: A Characterization of Data Provenance, in: Proceedings of the 8th International Conference on Database Theory, ICDT '01, Springer-Verlag, London, UK, UK, 2001, pp. 316–330. URL: http://dl.acm.org/citation.cfm?id=645504.656274.

[12] I. Foster, J. Vockler, M. Wilde, Y. Zhao, Chimera: A Virtual Data System for Representing, Querying, and Automating Data Derivation, Scientific and Statistical Database Management, International Conference on 0 (2002) 37.

[13] Y. L. Simmhan, B. Plale, D. Gannon, A survey of data provenance in e-science, SIGMOD Rec. 34 (2005) 31–36.

[14] Dublin Core Metadata Initiative, DCMI Metadata Terms, http://dublincore.org/documents/dcmi-terms/, 2012.

[15] A. Brazma, Minimum Information About a Microarray Experiment (MIAME)–successes, failures, challenges., The Scientific World Journal 9 (2009) 420–3.

[16] L. Moreau, B. Clifford, J. Freire, J. Futrelle, Y. Gil, P. Groth, N. Kwasnikowska, S. Miles, P. Missier, J. Myers, B. Plale, Y. Simmhan, E. Stephan, J. Van den Bussche, The Open Provenance Model core specification (v1.1), Future Generation Computer Systems 27 (2011) 743–756.

[17] Provenance Working Group W3C, 2011, Provenance Working Group W3C, URL: http://www.w3.org/2011/prov/wiki/Main_Page.

[18] J. Freire, D. Koop, E. Santos, C. T. Silva, Provenance for Computational Tasks: A Survey, Computing in Science Engineering 10 (2008) 11–21.

[19] C. Lim, S. Lu, A. Chebotko, F. Fotouhi, Prospective and Retrospective Provenance Collection in Scientific Workflow Environments., in: IEEE SCC, IEEE Computer Society, 2010, pp. 449–456. URL: http://dblp.uni-trier.de/db/conf/IEEEscc/scc2010.html#LimLCF10.

[20] P. T. Wood, Query languages for graph databases, ACM SIGMOD Record 41 (2012) 50.

[21] I. F. Cruz, A. O. Mendelzon, P. T. Wood, A graphical query language supporting recursion, in: Proceedings of the 1987 ACM SIGMOD international conference on Management of data, SIGMOD '87, ACM, New York, NY, USA, 1987, pp. 323–330. URL: http://doi.acm.org/10.1145/38713.38749. doi:10.1145/38713.38749.

[22] M. P. Consens, A. O. Mendelzon, Expressing structural hypertext queries in graphlog, in: Proceedings of the second annual ACM conference on Hypertext, HYPERTEXT '89, ACM, New York, NY, USA, 1989, pp. 269–292. URL: http://doi.acm.org/10.1145/74224.74247. doi:10.1145/74224.74247.

[23] R. H. Güting, GraphDB: Modeling and Querying Graphs in Databases, in: Proceedings of the 20th International Conference on Very Large Data Bases, VLDB '94, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1994, pp. 297–308. URL: http://dl.acm.org/citation.cfm?id=645920.672980.

[24] M. Gyssens, J. Paredaens, J. V. den Bussche, D. V. Gucht, A Graph-Oriented Object Database Model, IEEE Trans. Knowl. Data Eng. 6 (1994) 572–586.

[25] S. Abiteboul, D. Quass, J. McHugh, J. Widom, J. Wiener, The Lorel Query Language for Semistructured Data, Journal on Digital Libraries 1 (1996).

[26] M. Fernández, D. Fiorescu, A. Levi, D. Sucin, Declarative specification of Web sites with STRUDEL, The VLDB Journal 9 (2000) 38–55.

[27] P. Buneman, M. Fernandez, D. Suciu, UnQL: a query language and algebra for semistructured data based on structural recursion, The VLDB Journal 9 (2000) 76–110.

[28] G. Weikum, G. Kasneci, M. Ramanath, F. Suchanek, Database and information-retrieval methods for knowledge discovery, Commun. ACM 52 (2009) 56–64.

[29] S. Harris, A. Seaborne (eds), {SPARQL} 1.1 Query Language, Working Draft, W3C, 2010.

[30] T. Kifor, L. Z. Varga, J. Vazquez-Salceda, S. Alvarez, S. Willmott, S. Miles, L. Moreau, Provenance in Agent-Mediated Healthcare Systems, IEEE Intelligent Systems 21 (2006) 38–46.

[31] A. Benabdelkader, M. Santcroos, S. Madougou, A. H. C. van Kampen, S. D. Olabarriaga, A Provenance Approach to Trace Scientific Experiments on a Grid Infrastructure, in: eScience, IEEE Computer Society, 2011, pp. 134–141.

[32] A. Anjum, P. Bloodsworth, A. Branson, I. Habib, R. McClatchey, T. Solomonides, Research Traceability using Provenance Services for Biomedical Analysis, CoRR abs/1202.5 (2012).

[33] R. Hasan, R. Sion, M. Winslett, Introducing secure provenance: problems and challenges, in: Proceedings of the 2007 ACM workshop on Storage security and survivability, StorageSS '07, ACM, New York, NY, USA, 2007, pp. 13–18. URL: http://doi.acm.org/10.1145/1314313.1314318. doi:10.1145/1314313.1314318.

[34] TRANSFoRm, Translational Medicine and Patient Safety in Europe, 2010. URL: www.transformproject.eu.

[35] EHR4CR, Electronic Health Records for Clinical Research, 2011. URL: www.ehr4cr.eu.

[36] C. P. Friedman, A. K. Wong, D. Blumenthal, Achieving a Nationwide Learning Health System, Science Translational Medicine 2 (2010) 57cm29.

[37] J. Iavindrasana, A. Depeursinge, P. Ruch, S. Spahni, A. Geissbühler, H. Müller, Design of a Decentralized Reusable Research Database Architecture to Support Data Acquisition in Large Research Projects, in: MedInfo, 2007, pp. 325–329.

[38] V. Curcin, R. Danger, W. Kuchinke, S. Miles, A. Taweel, C. Ohmann, Provenance Model for Randomized Clinical Trials, in: Q. Liu, Q. Bai, S. Giugni, D. Williamson, J. Taylor (Eds.), Data Provenance and Data Management for eScience, volume 426 of *Studies in Computational Intelligence*, Springer, 2012.

[39] P. Missier, S. Dey, K. Belhajjame, V. Cuevas-Vicenttín, B. Ludäscher, D-PROV: extending the PROV provenance model with workflow structure, in: Proceedings of the 5th USENIX Workshop on the Theory and Practice of Provenance, TaPP '13, USENIX Association, Berkeley, CA, USA, 2013, pp. 9:1—-9:7. URL: http://dl.acm.org/citation.cfm?id=2482949.2482961.

[40] C. Bizer, D2RQ - treating non-RDF databases as virtual RDF graphs, in: In Proceedings of the 3rd International Semantic Web Conference (ISWC2004, 2004.

[41] Organization for the Advancement of Structured Information Standards, 2005, Security Assertion Markup Language (SAML) v2.0, URL: http://www.oasis-open.org/standards#samlv2.0.

[42] OPM, The Open Provenance Model, 2011. URL: openprovenance.org.

[43] W3C Working Group, Prov Implementations, 2013. URL: http://www.w3.org/2011/prov/wiki/ProvImplementations.

[44] J. F. Ethier, O. Dameron, V. Curcin, E. al., A Unified Structural/Terminological Interoperability Framework based on LexEVS: Application to TRANSFoRm, American Medical Informatics Association 20 (2013) 986–94.

[45] S. N. L. C. Keung, L. Zhao, E. Tyler, T. N. Arvanitis, Integrated Vocabulary Service for Health Data Interoperability, in: The Fourth International Conference on eHealth, Telemedicine, and Social Medicine (eTELEMED 2012), IARIA, Valencia, Spain, 2012, pp. 124–127. URL: /files/etelemed2012.pdf.

[46] S. Miles, P. Groth, S. Munroe, L. Moreau, PrIMe: A Methodology for Developing Provenance-Aware Applications, ACM Transactions on Software Engineering and Methodology 20 (2011) 1–42.

[47] P. Groth, S. Miles, W. Fang, S. C. Wong, K.-P. Zauner, L. Moreau, Recording and Using Provenance in a Protein Compressibility Experiment, in: Proceedings of the 14th IEEE International Symposium on High Performance Distributed Computing (HPDC'05), Research Triangle Park, North Carolina, 2005, pp. 201–208.

[48] J. Russell, R. Cohn, Neo4j, Book on Demand, 2012. URL: http://books.google.co.uk/books?id=Eb_GMQEACAAJ.

[49] A. P. Chapman, H. V. Jagadish, P. Ramanan, Efficient provenance storage, in: Proceedings of the ACM International Conference on Management of Data (SIGMOD), 2008, pp. 993–1006.

[50] E. Bertino, R. Sandhu, Database security – concepts, approaches, and challenges, IEEE TRANS. DEPENDABLE SECUR. COMPUT 2 (2005) 2–19.

[51] S. Kaushik, D. Wijesekera, P. Ammann, Policy-based dissemination of partial web-ontologies, in: Proceedings of the 2005 workshop on Secure web services, SWS '05, ACM, New York, NY, USA, 2005, pp. 43–52. URL: http://doi.acm.org/10.1145/1103022.1103030. doi:10.1145/1103022.1103030.

[52] B. Carminati, E. Ferrari, R. Heatherly, M. Kantarcioglu, B. Thuraisingham, A semantic web based framework for social network access control,

in: Proceedings of the 14th ACM symposium on Access control models and technologies, SACMAT '09, ACM, New York, NY, USA, 2009, pp. 177–186. URL: http://doi.acm.org/10.1145/1542207.1542237. doi:10.1145/1542207.1542237.

[53] A. DElia, J. Honkola, D. Manzaroli, T. Cinotti, Access Control at Triple Level: Specification and Enforcement of a Simple RDF Model to Support Concurrent Applications in Smart Environments, in: S. Balandin, Y. Koucheryavy, H. Hu (Eds.), Smart Spaces and Next Generation Wired/Wireless Networking, volume 6869 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 2011, pp. 63–74. URL: http://dx.doi.org/10.1007/978-3-642-22875-9_6.

[54] Q. Ni, S. Xu, E. Bertino, R. Sandhu, W. Han, An access control language for a general provenance model, in: Proceedings of the 6th VLDB Workshop on Secure Data Management, Springer Verlag, 2009, pp. 68–88. URL: http://www.springerlink.com/index/4V1X2374932725G3.pdf. doi:10.1007/978-3-642-04219-5_5.

[55] T. Cadenhead, V. Khadilkar, M. Kantarcioglu, B. Thuraisingham, A language for provenance access control, in: Proceedings of the first ACM conference on Data and application security and privacy, CODASPY '11, ACM, New York, NY, USA, 2011, pp. 133–144. URL: http://doi.acm.org/10.1145/1943513.1943532. doi:http://doi.acm.org/10.1145/1943513.1943532.

[56] M. K. Tyrone Cadenhead, B. Thuraisingham, A Framework for Policies over Provenance, in: 3rd USENIX workshop on the Theory and Practice of Provenance, 2011.

[57] U. Braun, A. Shinnar, M. Seltzer, Securing Provenance, in: The 3rd USENIX Workshop on Hot Topics in Security, USENIX HotSec, USENIX Association, Berkeley, CA, USA, 2008, pp. 1–5.

[58] O. Sacco, A. Passant, S. Decker, An Access Control Framework for the Web of Data, in: Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on, 2011, pp. 456–463. doi:10.1109/TrustCom.2011.59.

[59] B. Stepien, S. Matwin, A. P. Felty, Advantages of a non-technical XACML notation in role-based models, in: PST, 2011, pp. 193–200.

[60] N. Helil, K. Rahman, Extending XACML profile for RBAC with semantic concepts, in: Computer Application and System Modeling (ICCASM), 2010 International Conference on, volume 10, 2010, pp. V10–69 –V10–74. doi:10.1109/ICCASM.2010.5622888.

[61] G. Kounga, M. C. Mont, P. Bramhall, Extending XACML access control architecture for allowing preference-based authorisation, in: Proceedings of the 7th international conference on Trust, privacy and security in digital business, TrustBus'10, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 153–164. URL: http://dl.acm.org/citation.cfm?id=1894888.1894907.

[62] S. Dey, D. Zinn, B. Ludaescher, Publishing Privacy-Aware Provenance by Inventing Anonymous Nodes, in: Proceedings of the Fourth International Workshop on REsource Discovery (RED 2011), 2011.

[63] S. Dey, D. Zinn, B. Ludäscher, ProPub: Towards a Declarative Approach for Publishing Customized, Policy-Aware Provenance, in: J. Bayard Cushing, J. French, S. Bowers (Eds.), Scientific and Statistical Database Management, volume 6809 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 2011, pp. 225–243. URL: http://dx.doi.org/10.1007/978-3-642-22351-8_13.

[64] R. Danger, R. Joy, J. Darlington, V. Curcin, Provenance and Annotation of Data and Processes, in: P. Groth, J. Frew (Eds.), Lecture Notes in Computer Science: Provenance and Annotation of Data and Processes, volume 7525, Springer Berlin Heidelberg, Berlin, 2012, pp. 233–35.

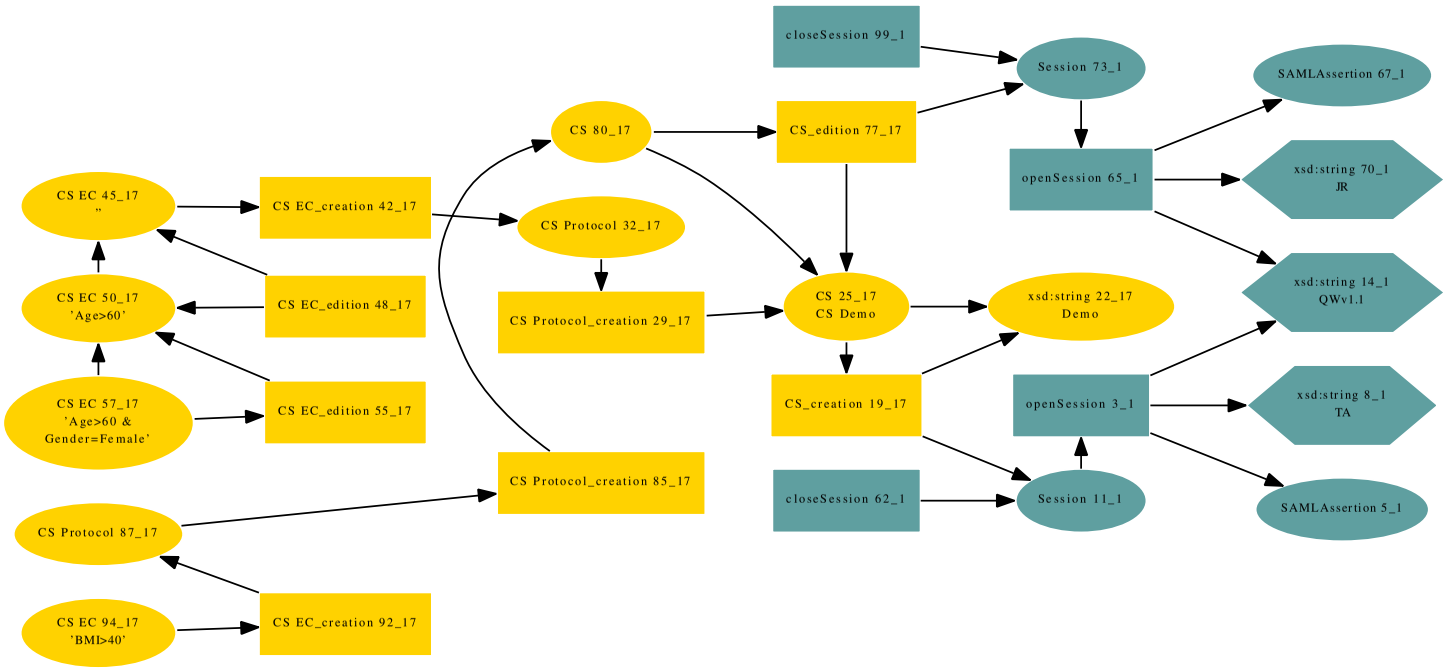

Figure 3: Provenance graph conforming to the template in Figure 2 contains the trace of the evolution of a Clinical Study entity 'CS Demo' during the study preparation phase. Two versions of the entity are shown in the graph: 25_17 and 80_17, each one defining a protocol along with its eligibility criteria. Both versions were created using the same software tool, TRANSFoRm Query Workbench v1.1 (agent 14_1 in the graph) during two editing sessions opened by two different users (JR and TA). First version, 25_17, had a Protocol created with Eligibility Criteria, initially empty (CS EC 45_17) and then edited twice to specify recruitment condition Age $>$ 60 & Gender = Female. Another protocol (87_17) was created with eligibility criteria BMI $>$ 40 and resulted in the new version of Clinical Study (CS 80_17). NB: CS and EC are label acronyms for ontological concepts Clinical Study and Eligibility Criteria in RCTPO, colours denote different software tools that contributed different graph fragments, yellow for Query Workbench, blue for Authentication Framework.

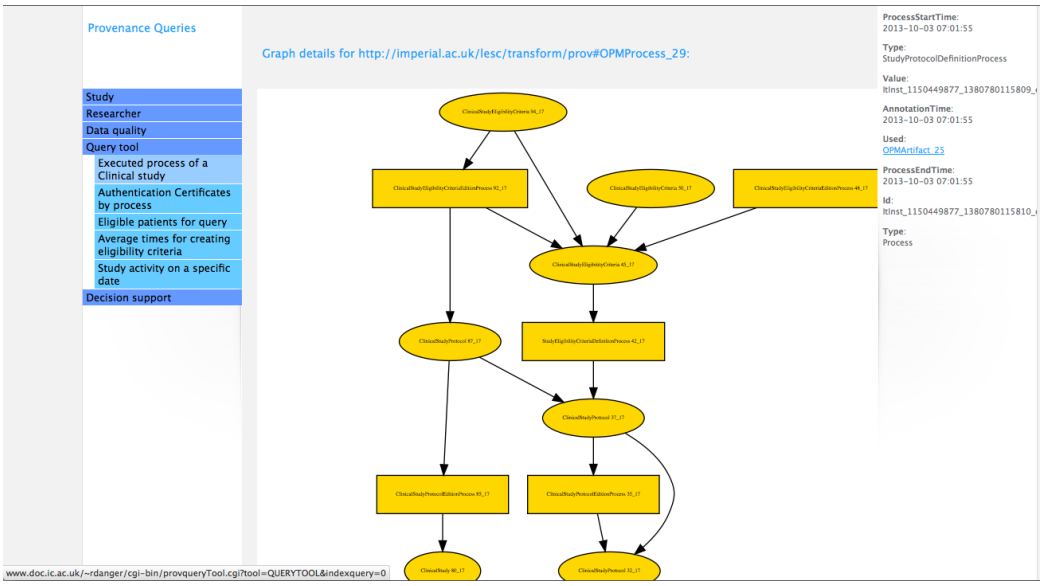

Figure 4: TRANSFoRm query tool showing the graph area surrounding a selected item - StudyProtocolDefinitionProcess. The menu bar on the left contains configured queries.

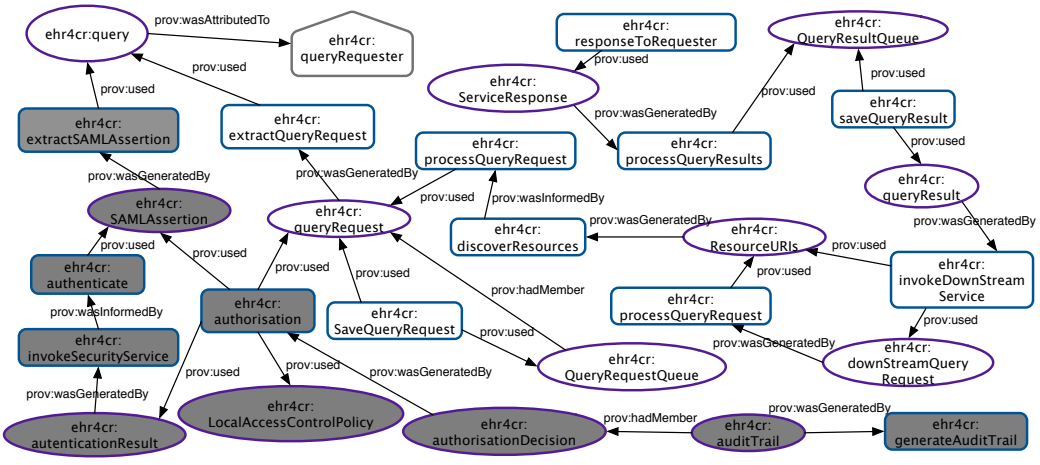

Figure 5: A provenance recording template for integrated auditing

1. Syntax Use W3C PROV or the Open Provenance Model for modelling provenance data recorded in an electronic healthcare system. When choosing between the two, consider available tools and libraries in both systems for potential reuse or adaptation.
2. Vocabulary Institutions involved in a distributed healthcare application should agree to a common vocabulary for describing provenance entities where possible.
3. Domain models Link provenance models and data with domain knowledge models and data, respectively. These may be expressed in the form of templates, similar in form to workflows or business processes, and should be shared with the community.
4. Detail focus Always aim to model the detail of what happens, including each processing step and data item involved, rather than summary information that directly answers current provenance questions.
5. Granularity Use the existing biomedical models and ontologies as indicators of the level of granularity of process description that users are likely to be interested in the future, and validate it against the provenance questions known at design time.
6. Connectivity Model the provenance with the expectation that, if brought together, it would form a single graph describing the full, interconnected history of the system, as opposed to being delimited into a set of isolated records. Globally unique IDs should be used where feasible to facilitate interlinking.
7. Human actions Include both salient human activities and automated processes in the model of provenance to be captured.
8. Model elements In designing the provenance model, consider explicitly representing each element of the process flows and the data flows, as well as the attributions of actions to users.

Table 1: Overview of recommendations (1)

9. Reuse of existing data Where feasible, integrate the data captured by existing mechanisms (version control, audit, logging, etc.) into the provenance record, whether by reference or translation, to provide as rich and integrated an account as possible.
10. Library reuse If not using shared execution middleware with provenance capture support, make use of existing OPM/PROV libraries to capture provenance data within your application code.
11. Timely capture Build the runtime provenance capture functionality into each step of the system processes, at the appropriate level of granularity, using global identifiers where possible and linking to records of preceding steps using the chosen provenance model relations.
12. Performance testing Test the performance overhead of provenance capture within a small sample part of your application to ensure it is acceptable for your application or if the technologies used and level of detail of model need to be modified.
13. Database structure Consider whether provenance storage needs to be provided in a relational database, if one is the standard solution in the research environment, or would an RDF store or a NoSQL graph database be an option. Privacy and data regulations will play an important role in deciding whether and how best to distribute the data in a multiple site scenario.
14. Data infrastructure Reuse existing application infrastructure for provenance transmission if storing provenance centrally rather than at each site. Messaging systems may be used to transmit data reliably and asynchronously between application sites.
15. Rate of growth Due to the typically large size of provenance data collected over time, it is crucial to establish early on the rate at which provenance storage requirements are expected to increase over time. The level of detail in the model may need to be adjusted accordingly and some deletion/archiving procedures introduced.
16. Permissions When implementing provenance access control, consider the differences in access permissions between concrete data, and provenance records of that data.

Table 2: Overview of recommendations (2)

17. Restrictions When designing an access control mechanism for provenance data, decide whether restricted provenance information should be completely hidden or just have details abstracted. If the latter is the case, a mechanism is needed for answering user queries using graphs restricted to that user's access level.
18. Sensitive data When provenance is used to capture data about the actions of the security layers, separation of functionality needs to be introduced to avoid inadvertently exposing sensitive data. One way of doing so is by providing token identifiers which can then be used to request detail from the security mechanism.
19. Extensible queries Introduce mechanisms for dynamically adding new provenance queries to the software.
20. Interactive querying Provenance data can be used to answer individual queries or create tabular and graphical reports, but added value can be obtained by exploiting the graph representation of provenance to support open-ended, investigative querying.
21. Breaking abstraction When creating provenance queries, whether for atomic questions, or for navigating the provenance graph space, try to find queries that can be optimised by being directly formulated in the underlying query language and investigate whether the performance gain warrants this.

Table 3: Overview of recommendations (3)