Enhancing Workflow with a Semantic Description of Scientific Intent

Edoardo Pignotti[a], Peter Edwards[a], Nick Gotts[b], Gary Polhill[b]

[a]*Computing Science, University of Aberdeen, Aberdeen AB24 5UE, UK*
[b]*The Macaulay Institute, Ceaigirbuckler, Aberdeen AB15 8QH, UK*

Abstract

Scientists are becoming increasingly dependent upon resources available through the Internet including, for example, datasets and computational modelling services, which are changing the way they conduct their research activities. This paper investigates the use of workflow tools enhanced with semantics to facilitate the design, execution, analysis and interpretation of workflow experiments and exploratory studies. Current workflow technologies do not incorporate any representation of experimental constraints and goals, which we refer to in this paper as scientist's intent. This paper proposes an abstract model of intent based on the Open Provenance Model (OPM) specification. To realise this model a framework based upon a number of Semantic Web technologies has been developed, including the OWL ontology language and the Semantic Web Rule Language (SWRL). Through the use of social simulation case studies the paper illustrates the benefits of using this framework in terms of workflow monitoring, workflow provenance and annotation of experimental results.

Keywords: workflow, semantic grid, provenance, scientist's intent, rules

1. Introduction

Collaborations between large groups of scientists are increasingly seen as essential in order to address the complex challenges facing society [1]. Researchers rely extensively on computer and communication technologies to bring together different expertise, using the Web as a convenient vehicle of communication. This paper investigates the issues related to the documentation of scientific processes and their products in order to support reproducibility, scientific discovery and result interpretation.

A number of so-called e-Science[1] activities have focused on facilitating and promoting collaboration between scientists using advanced distributed information management systems [2]. The original vision of e-Science was to facilitate large scale science using Grid technologies [3] as a fundamental computing infrastructure to manage distributed computational resources and data. However, a major gap exists between current technologies and the original vision of e-Science. Where Grid technologies overcome some of the limitations of existing Web tools in terms of managing computational tasks, there is still a need for technologies to support truly flexible collaboration. For these reasons the concept of a Semantic Grid [4] emerged, which integrates Semantic Web [5] and Grid technologies.

A number of scientific Grid infrastructures have emerged in recent years, such as the UK National Grid Service[2] (NGS), Distributed European Infrastructure for Supercomputing Applications[3] (DEISA), Enabling Grids for e-Science[4] (EGEE), and Open Science Grid[5] (OSG). These are changing the way in which research is conducted with increasing emphasis on 'in silico' experiments as a way to test hypotheses. Scientific workflow technologies [6] have emerged to allow researchers to create and execute experiments given a pool of available services in such environments [7, 8, 9, 10, 11]. These languages and tools are designed to capture the flow of information between services (e.g. service addresses and relations between inputs and outputs). However, in order to fully characterise scientific analysis it is necessary to go beyond such low-level descriptions by capturing the experimental conditions.

Openness and accountability is an important aspect of science and proper documentation of scientific activities is therefore essential in order to understand and reproduce experimental processes. As scientific workflows become ever more complex, so does the problem of workflow documentation and management of provenance information. Provenance is increasingly seen as an essential aspect of scientific workflow in order to support result interpretation, problem diagnosis and scientific discovery [12, 13]. Provenance (also referred to as lineage or heritage) aims to provide additional documentation about the processes that led to the creation of a resource [14]. [15] expands on the Zachman Framework [16] by presenting the '7 W's of Provenance': *Who, What, Where, Why, When, Which, & (W)How*. While some progress has been made in terms of documenting processes (*Who, What, Where, When, Which, &*

Email addresses: e.pignotti@abdn.ac.uk (Edoardo Pignotti), p.edwards@abdn.ac.uk (Peter Edwards), n.gotts@macaulay.ac.uk (Nick Gotts), g.polhill@macaulay.ac.uk (Gary Polhill)

[1] http://www.rcuk.ac.uk/escience

[2] http://www.ngs.ac.uk/

[3] http://www.deisa.eu/

[4] http://www.eu-egee.org/

[5] http://www.opensciencegrid.org/

(W)How), little effort has been devoted to capturing the *Why* aspect of research methodology.

The *International Provenance and Annotation Workshop* series (IPAW06, IPAW08)[6] has investigated issues related to data provenance and process documentation which has lead to the development of an open model for describing provenance. The Open Provenance Model (OPM) [17] was developed to address issues in managing provenance information in workflow-driven science. OPM is designed to make assertions about causation between processes and data and between processes and agents. However, the OPM model does not reflect that scientists (agents) can make decisions about the experiment that they are conducting based on their intent and that such decisions can influence how the processes are executed.

Our work aims to make the constraints and goals of a scientific workflow, which we describe as the *scientist's intent*, transparent. Existing workflow languages are unable to convey such intent information because they are designed to capture low-level service composition rather than higher-level descriptions of the experimental process. We argue that a representation of *scientist's intent* can provide the following benefits: (1) reduced human effort in inspecting workflow documentation (e.g. checking constraints and ensuring goals are met); (2) better management of workflow execution.

The remainder of this paper is organised as follows:

In Section 2, we present a social simulation case study that illustrates how agent-based simulation can be supported by the use of workflow technologies. The case-study is used throughout this paper to highlight some of the limitations of current workflow technologies and to illustrate the issues that need to be addressed in order to make workflow experiments more transparent. In Section 3, we describe the formal representation of the Open Provenance Model and discuss the challenges that we face in order to represent scientist's intent using the existing OPM representation. Section 4 introduces an extended version of the Open Provenance Model capturing aspects of scientist's intent. Section 5 describes an OWL[7] binding of our scientist's intent model which combines an existing OWL realisation of OPM with SWRL-based[8] rules and a scientist's intent ontology. Section 6 discusses a semantic workflow architecture based on an extension of Kepler and a number of Semantic Grid and Web services. In Section 7, we present the result of an evaluation of the new intent formalism, realisation and implementation. Section 8 covers related work, in particular existing workflow languages and systems, provenance frameworks and scientific workflow applications and identifies the limitations of current workflow documentation solutions. Finally, in Section 9 we present our conclusions and possible future directions.

2. Simulation Case Study

One of the main challenges in simulation is the need to improve the scientific rigour of agent-based modelling. An agent-based model (ABM) is a class of computational models for simulating the actions and interactions of autonomous agents. An important aspect of science is that work should be repeatable and verifiable. In this context, workflow technologies can facilitate the design, execution, analysis and interpretation of simulation experiments and exploratory studies. However, current workflow technologies can only capture the method and not the scientist's intent which we have argued [18] is essential to making the experiment truly transparent.

2.1. Biodiversity Scenario

We now present an analysis of the benefits and limitations of workflow technologies in describing scientist's intent by exploring a simulation case study where workflow technologies were used to automate a simulation experiment. The focus of the case study is on investigating the effects of various agri-environment incentive schemes on biodiversity. The specific research question is whether there is any difference between the effectiveness of activity- or outcome-based incentive schemes in maintaining what here we call, for brevity, biodiversity, i.e. the number of species and their persistence time before extinction. To study these issues the FEARLUS model [19] has been coupled with an ecological metacommunity model SPOMM (Stochastic Patch Occupancy Metacommunity Model [20]).

2.2. Methodology

We are interested in the range of dynamics that the FEARLUS-SPOMM model can generate using each incentive scheme, which arises more from the relationships among the parameters than their absolute values, and the consequences for biodiversity. The approach being taken is as follows:

- Perform an initial interactive exploration of parameter space to find the range of dynamics the model is capable of producing, and identify a set of parameters to explore with more rigorous experiments;
- Identify outcomes that are unrealistic given empirical evidence (e.g. an annual bankruptcy rate of more than 5%) and reject permutations of parameters generating these outcomes;
- Identify outcomes of interest (e.g. runs where the habitat for a certain species drops below a specified level in any one year);
- Use batch runs of the model to explore the parameter permutations identified;
- Build a regression tree of the results and inspect whether or not it contains the incentive scheme as an explanatory variable.

The workflow shown in Figure 1 is designed to perform the parameter exploration phase of the biodiversity case study. A range of possible combinations of parameters are explored (e.g. combinations of Activity-Species Reward Ratio, Aspiration Threshold, etc.). The generation of parameters (Generate

[6] http://www.ipaw.info/
[7] http://www.w3.org/2004/OWL
[8] http://www.w3.org/Submission/SWRL/

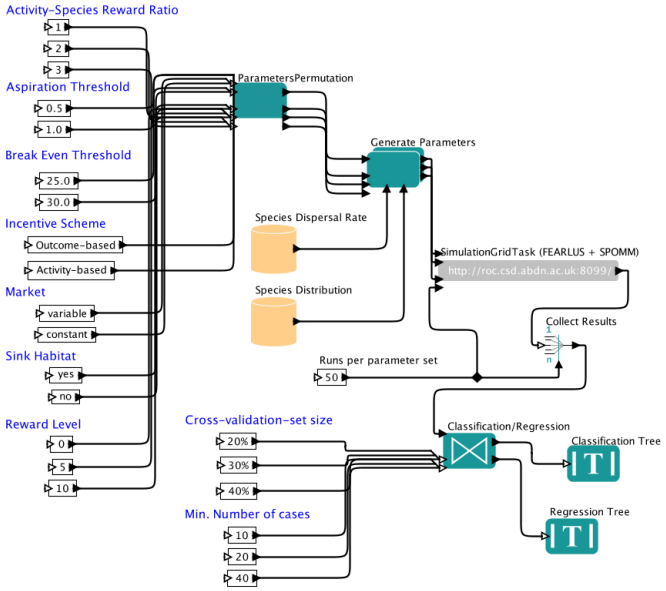


Figure 1: Example Workflow for the FEARLUS-SPOMM Case Study (Parameter Exploration).

Parameters activity) is based on the combination of parameters set generated by the permutation activity and real data (Species Dispersal Rate and Species Distribution). As many runs as possible are carried out by the Simulation Grid Task (e.g. 50). Such workflow activity[9] distributes simulation runs across computing hosts available on the Grid. The results of all runs are than collected and used to generate a set of classification/regression trees. Different combinations of parameters are used for the classification/regression algorithm (e.g. Cross-validation-set size and Min. Number of cases per Node).

2.3. Issues and Requirements

During the investigation of this case study we encountered several issues related to workflow documentation that need to be addressed in order to satisfy the requirements posed by the methodology used in this simulation experiment. The general issue applies to scientific workflows and is related to the workflow not being able to capture the goals and constraints associated with an experiment. Such goals and constraints are very important in order to understand the intent of the scientists while running a particular experiment. We recognise that understanding the scientist's intent is essential in order to make a simulation experiment truly transparent.

For example, in the workflow in Figure 1 it is not clear from the workflow itself what the goal of this experiment is.

Goal 1. The goal of this experiment is to obtain at least one cross-validated tree that explains 5% of the deviation of the result data.

The subgoal of the experiment is:

Goal 2. To obtain a similar number of valid runs between outcome-based and activity-based incentive schemes.

The researcher might also have constraints associated with the experiment.

Constraint 1. If a simulation run has a bankruptcy rate of more than 5%, drop the entire parameter set.

Constraint 2. If any habitat type, at any time step, drops below a specified threshold, alert the user.

Constraint 3. If a tree does not identify an Incentive Scheme as an influential variable, alert the user.

Constraint 4. If in a simulation run, one land manager owns more than half of the land, the entire simulation can be discarded.

Constraint 5. If the platform is not compatible with IEEE 754[10] the test might be invalid as this could change the results of the comparison test.

We have illustrated through the use of a biodiversity case study how agent-based simulation can be supported by the use of workflow technologies. Although the workflow metaphor is important in terms of facilitating the transparency and repeatability of simulation experiments, we have explained our concern that the goals and constraints associated with such experiments cannot be sufficiently described in a workflow.

The workflow, goals and constraints are used as examples to illustrate the design, realisation and implementation of a *scientist's intent* model. Moreover, these examples are used as a basis for the evaluation of the *scientist's intent* model and its implementation.

In order to summarise the requirements emerged from this case-study we now present a set of intent queries [11]:

- (CS1) What was the intent of the scientists while running this experiment?
- (CS2) Have the goals of this experiment been achieved?
- (CS3) Did the experiment violate any of the constraints defined by the scientists?
- (CS4) What decisions were made while executing this experiment?

3. Capturing the Provenance of an Experiment

Using workflow technologies to perform scientific experiments has a significant advantage in terms of annotating, discovering and reasoning about processes and data [21, 22]. Information on the computational outcome of a scientific workflow, the data and the processes involved are essential in order

[9] A workflow activity is an implementation of the behaviour of a particular operational step of a workflow.

[10] http://grouper.ieee.org/groups/754/

[11] We acknowledge that the queries presented here are extracted from the specific case-studies investigated in our work and that there may be more queries related to intent.

to understand an experiment, and it is therefore important to provide an accurate account of the provenance of a workflow experiment.

In this section we introduce the formal representation of the Open Provenance Model and discuss the challenges that we face in order to represent scientific intent information using the existing OPM specification.

3.1. An Open Provenance Model

The Open Provenance Model [17] is an abstract model which provides a specification to express data provenance, process documentation and data derivation. OPM is designed to enable systems to exchange information about processes and data artifacts without forcing semantics and schema for the data. The OPM defines a model based on three kind of nodes:

- *Artifact*: Immutable piece of state, which may have a physical embodiment in a physical object, or a digital representation in a computer system;
- *Process*: Action or series of actions performed on or caused by artifacts, and resulting in new artifacts;
- *Agent*: Contextual entity acting as a catalyst of a process, enabling, facilitating, controlling, or affecting its execution.

And a simple semantics of observation and causation based on five causal relationships:

- a process *used* an Artifact;
- an Artifact *was generated by* a process;
- a process *was triggered by* a process;
- an Artifact *was derived from* an Artifact;
- a process *was controlled by* a agent

The combination of the nodes and the relationships allow the creation of causality graphs. OPM allows for different accounts of "past execution" to be represented in the same graph therefore offering different levels of explanation of such executions. Figure 2 shows an example of a provenance account using OPM representing a baking process controlled by John (the agent) which used flour, butter, sugar and eggs (input artifacts) to bake a cake (output artifact).

The OPM also defines completion rules that can be applied to the provenance graph. For example the causal relation *wasTriggeredBy* can be inferred from the existence of a *used* and *wasGeneratedBy* relation.

The Open Provenance Model allows for indirect relationships to be inferred by providing multi-step versions of the existing relationships, namely *Used**, *WasTriggeredBy**, *WasDerivedFrom** and *WasTriggeredBy**. Such relationships are used in order to find the causes of an artifact or a process involving multiple transitions, for example the artifact a_1 was derived from artifact a_2 possibly using multiple steps.

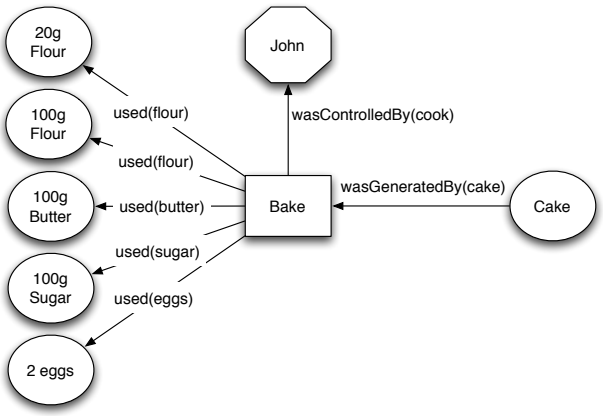


Figure 2: An Example of Provenance Graph (adapted from [17]).

Provenance is perceived as an important component of workflow systems to provide documentation about scientific analyses and processes. Provenance documentation is particularly important in the scientific method in order to understand and reproduce such processes. In the introduction we discussed the need to capture the scientist's intent associated with a workflow in order to better characterise an experiment. As provenance should enable users to understand, verify, reproduce, and ascertain the quality of data products generated by processes we argue that intent information should be a kay element of the provenance documentation.

As stated above, the Open Provenance Model is designed to make assertions about causation between processes and data and between processes and agents. However, OPM does not take into account the fact that agents can make decisions about the processes that they are controlling based on intent and that they can influence how the process is executed as a result of this. To better understand this concept let us consider the provenance example presented in [17]. Assuming the agent "John" is driven by a goal ("bake a cake of an acceptable quality") and has a constraint ("the mix must have an hydration level between 50% and 60%"), the resulting provenance graph may be as shown in Figure 2. However, if an additional 20g of flour was used by "John" during the baking process as a result of violating the constraint, it is not possible to understand from Figure 2 why this was the case.

In intelligent software agents, goals and constraints are introduced to explain and specify an agent's proactive behaviour. In this view, agents are assumed to have their own goals which initiate behaviour that can be influenced by certain constraints [23]. Such concepts resonate very well with the idea of intent and the concept of an agent in OPM. For example, a scientist (*Agent*) can make decisions (*Action*) about the experiment he is conducting (*Process*) based on certain desires about the experiment (*Scientist's Intent*).

In order to answer the intent queries presented in the case-study section we need to incorporate the concept of intent into the provenance graph. This will allow us to address specific queries based on the OPM formalism such as the following:

- (Q1) What was the intent of agent Ag_x?
- (Q2) What was the intent of agent Ag_x while controlling a process P_x?
- (Q3) What decisions has agent Ag_x made while controlling a process P_x?
- (Q4) Were any constraints defined by the agent Ag_x violated?
- (Q5) Did an agent Ag_x achieved his goal/goals?
- (Q6) What are the decisions made by agent Ag_x that have influenced an artifact A_x?

4. Scientist's Intent

In Section 3 we discussed how the Open Provenance Model does not take into account agents making decisions about the processes that they are controlling. In this Section, we introduce an extended version of the Open Provenance Model capturing aspects of a scientist's intent. An OWL binding of the extended OPM model is also presented that combines an OWL realisation of OPM (including SWRL rules) and a scientist's intent ontology.

4.1. An Abstract Model of Scientist's Intent

We have introduced a minimal set of concepts providing a formal representation of a scientist's intent based on an extension of the Open Provenance Model. In this model, we take the view that agents are driven by intent and they can therefore make decisions about the processes that they are controlling. Our scientist's intent model is based on four entities that we define below:

Definition 1. (State) A kind of *artifact* describing the internal properties of a *process*.

Definition 2. (Intent) An anticipated outcome that guides the *agent*'s planned actions based on a set of goals and constraints.

Definition 3. (Goal) A specification of a desired state of the process (e.g. properties associated with the state) that an *agent* is currently controlling.

Definition 4. (Constraint) A specification of a restriction on the properties associated with the state of a *process*.

Definition 5. (Decision) A course of action based on the consideration of a goal or a constraint.

In order to capture the causal dependencies between the entities presented above and the entities defined by OPM (*Agent*, *Artifact* and *Process*), we introduce five causal relationships:

Definition 6. (WasDrivenBy) An edge "was driven by" between an agent Ag_1 and an intent In_1 indicates that the agent was driven by the intent defined by I_1.

Definition 7. (WasMadeBy) A connection between a decision De_1 and an agent Ag_1 by a "was made by" edge indicates that the decision De_1 was made by the agent Ag_1 .

Definition 8. (WasInfluencedBy) The assertion of an edge "was influenced by" between a process P_1 and a decision De_1 or between an artifact A_1 and a decision De_1 indicates respectively that the process P_1 was influenced by the decision De_1 or the artifact A_1 was influenced by the decision De_1.

Definition 9. (Shaped) An edge "shaped" between a goal Gl_1 and an intent In_1 or a constraint Co_1 and an intent In_1 indicates that the goal Gl_1 or the constraint Co_1 has shaped the intent In_1.

Definition 10. (WasBasedOn) The assertion of an edge "was based on" between a decision De_1 and a goal Gl_1 or constraint Co_1 indicates that the decision De_1 was influenced by the goal Gl_1 or the constraint Co_1.

An overview on this model is presented in Figure 3.

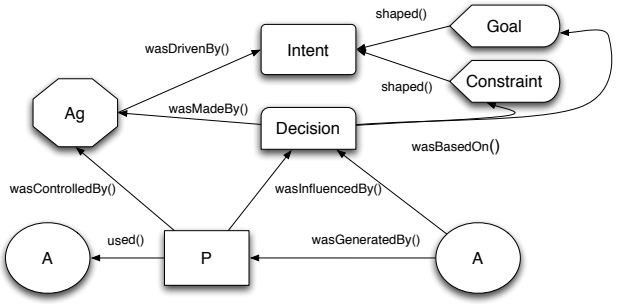


Figure 3: An Abstract Model of Scientist's Intent Based on the extended OPM Specification.

We expand the OPM provenance graph definition with the the following rules:

- *Intent*, *Goal*, *Constraint* and *Decision* entities are defined by unique identifiers belonging respectively to the set of *IntentId*, *GoalId*, *ConstraintId* and *DecisionID*.
- *in* represents the ID of an intent, *gl* represents the ID of a goal, *co* represents the ID of a constraint and *de* represents the ID of a decision.
- Functions *In*, *Gl*, *Co* and *De* are defined in order to access information about specific intents, goals, constraints and decisions.
- An OPM graph is defined in our model as $gr = \langle A, P, AG, U, G, T, D, C, Ov, Re, In, Gl, Co, De, Db, Mb, Ib, Df, Bo \rangle$ where $A \in Artifact$, $P \in Process$, $AG \in Agent$, $U \subseteq Used$, $G \subseteq WasGeneratedBy$, $T \subseteq WasTriggeredBy$, $D \subseteq WasDerivedFrom$, $C \subseteq WasControlledBy$, $Ov \subseteq Overlapping$, $Re \subseteq Refinement$, $In \in Intent$, $Gl \in Goal$, $Co \in Constraint$, $De \in Decision$, , $Db \subseteq WasDrivenBy$, $Mb \subseteq WasMadeBy$, $Ib \subseteq WasInfluencedBy$, $Df \subseteq Shaped$, $Bo \subseteq WasBasedOn$.

- Structural equality applies to the edges introduced in this model: two edges *wasDrivenBy*, *wasMadeBy*, *wasInfluencedBy*, *Shaped*, *wasbasedOn* are equal if they have the same source, the same destination and the same accounts.

Figure 4 shows the extensions introduced to the OPM model by scientist's intent.

$$
\begin{aligned}
\textbf{IntentId} &: \textit{primitive set}\\
\textbf{DecisionId} &: \textit{primitive set}\\
\textbf{GoalId} &: \textit{primitive set}\\
\textbf{ConstraintId} &: \textit{primitive set}\\
\textbf{Intent} &= \textit{IntentId} \rightarrow \textit{Value}\mathbb{P}(\textit{Account})\\
\textbf{Decision} &= \textit{DecisionId} \rightarrow \textit{Value}\mathbb{P}(\textit{Account})\\
\textbf{Goal} &= \textit{GoalId} \rightarrow \textit{Value}\mathbb{P}(\textit{Account})\\
\textbf{Constraint} &= \textit{ConstraintId} \rightarrow \textit{Value}\mathbb{P}(\textit{Account})\\
\textbf{WasDrivenBy} &= \textit{Agent} \times \textit{Role} \times \textit{Intent} \times \mathbb{P}(\textit{Account}) \times \textit{OTime}^0\\
\textbf{WasMadeBy} &= \textit{Decision} \times \textit{Role} \times \textit{Agent} \times \mathbb{P}(\textit{Account}) \times \textit{OTime}^0\\
\textbf{WasInfluencedBy} &= \textit{Process} \times \textit{Artifact} \times \textit{Decision} \times \mathbb{P}(\textit{Account}) \times \textit{OTime}^0\\
\textbf{WasBasedOn} &= \textit{Decision} \times \textit{Goal} \times \textit{Constraint} \times \mathbb{P}(\textit{Account}) \times \textit{OTime}^0\\
\textbf{Shaped} &= \textit{Intent} \times \textit{Goal} \times \textit{Constraint}\\
\textit{OPMGraph} &= \textit{Artifact} \times \textit{Process} \times \textit{Agent} \times \textbf{Intent} \times \textbf{Decision}\\
&\quad \times \textbf{Goal} \times \textbf{Constraint} \quad \times \mathbb{P}(\textit{Used}) \times \mathbb{P}(\textit{WasGeneratedBy})\\
&\quad \times \mathbb{P}(\textit{WasTriggeredBy}) \times \mathbb{P}(\textit{WasDerivedFrom})\\
&\quad \times \mathbb{P}(\textit{WasControlledBy}) \times \mathbb{P}(\textit{Overlaps})\\
&\quad \times \mathbb{P}(\textbf{WasDrivenBy}) \quad \times \mathbb{P}(\textbf{WasMadeBy})\\
&\quad \times \mathbb{P}(\textbf{WasInfluencedBy}) \times \mathbb{P}(\textbf{WasBasedOn}) \times \mathbb{P}(\textbf{Shaped})
\end{aligned}
$$

Figure 4: An Extension of the OPM Causality Graph Data Model.

Figure 5 presents an alternative account of the provenance example described in Figure 2. We now demonstrate through this example that the set of concepts introduced in this section are minimal. In this example, we are showing an alternative derivation of the process "baking" using our model of intent. In this provenance account, "John" (the agent) is driven by the intent defined by a goal "bake a cake of an acceptable quality" and a constraint "if the mix is too runny, you need to add more flour". Without the concept of *Intent*, *Goal* and *Constraint* it will not be possible to describe the intent behind the agent. Moreover, the use of the relationships *shaped* and *wasDrivenBy* allow the correlation between the goal, constraints and intent and the agent. In order to represent a *Goal* it is also necessary to characterise the *State* of a *Process* as a goal is a specification of a desired state, e.g. cake of an acceptable quality.

Figure 5 illustrates that while adding the flour to the cake mix ("Add Flour to Mix" process), the agent "John" made the decision to add an additional 20g of flour based on the goal and constraints defined in his intent. In this case, the concept of *Decision* is required in order to justify the additional 20g of flour added into the mix.

Comparing the example in Figure 2 with the account of provenance in Figure 5 using our model of intent we can infer the following:

- The agent John was driven by the intent: goal "bake a cake of an acceptable quality", constraint "if the mix is too runny, you need to add more flour", (requires Goal, *Constraint*, *Intent*, *Shaped* and *wasDrivenBy*);
- The existence of the "20g Flour" artifact is influenced by the agent's decision to add more flour to the mix (requires *Decision* and *wasMadeBy*).
- The decision to add more flour to the mix was based on the goals and constraints defined in the agent's intent (requires *wasBasedOn*).
- The"Add Flour to Mix" process and the"Cake Mix" artifact were influenced by the decision to add more flour (requires *wasInfluencedBy*).

4.1.1. Inference Rules

We now introduce a set of inference rules that can be performed on our extended version of the Open Provenance Model.

If an artifact a_1 was influenced by a decision de_1 we can infer that if a process p_1 used an artifact a_1 it was also influenced by the decision de_1 as presented by definition 11.

Definition 11. (WasInfluencedBy) A process p_1 was influenced by a decision de_1 if the artifact a_1 used by the process p_1 was influenced by de_1.

If a process p_1 was influenced by a decision de_1 it is unreasonable to infer that an artifact a_1 was also influenced by the decision de_1 unless implicitly defined in the provenance graph; however it is useful to infer that such relationship may exist (see example in Figure 6). We introduce a new edge *MayHaveBeenInfluencedBy* and an definition (Definition12) to define such a relationship.

Definition 12. (MayHaveBeenInfluencedBy) A process p_1 may have been influenced by a decision de_1 if the artifact a_1 generated by the process p_1 was influenced by de_1.

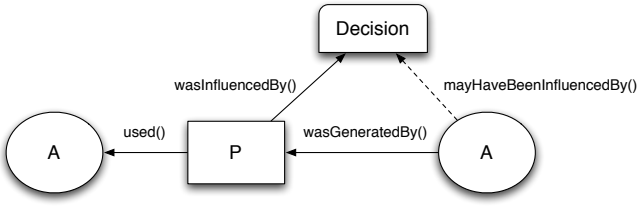


Figure 6: Example of *MayHaveBeenInfluencedBy* Relationship Inferred About an *Artifact*.

The *MayHaveBeenInfluencedBy* relationship only works if there is direct causality between a *Process* or an *Artifact* and a *Decision*. However, we are also interested to find out the decisions that may influence an *Artifact* or a *Process* based on their relationship with other *Artifacts* or *Processes* using multiple steps.

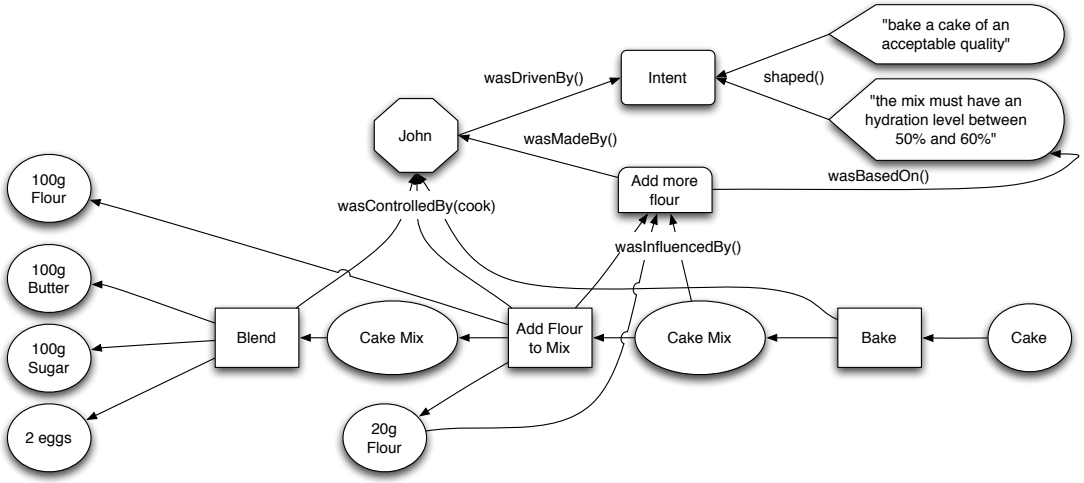


Figure 5: An Alternative Account of the Provenance Presented in Figure 2 Using our Model of Intent.

The Open Provenance Model allows for indirect causality relationships to be inferred by providing transitive versions of the basic causal relationships namely *Used**, *WasTriggeredBy**, *WasDerivedFrom** and *WasTriggeredBy**. We introduce a transitive version of the *mayHaveBeenInfluencedBy* relationship (namely *MayHaveBeenInfluencedBy**) which is described by definitions 13, 14, 15 and 16.

Definition 13. (Multi-Step MayHaveBeenInfluencedBy) A process p may have been influenced by a decision *de* (possibly using multiple steps) written as $p \rightarrow^* de$, if the process p used (possibly using multiple steps) artifact a and a may have been influenced by decision *de*.

Definition 14. (Multi-Step MayHaveBeenInfluencedBy) A process p may have been influenced by a decision *de* (possibly using multiple steps) written as $p \rightarrow^* de$, if the process p was triggered by (possibly using multiple steps) process p_1 and p_1 may have been influenced by decision *de*.

Definition 15. (Multi-Step MayHaveBeenInfluencedBy) An artifact a may have been influenced by a decision *de* (possibly using multiple steps) written as $a \rightarrow^* de$, if the artifact a was generated by (possibly using multiple steps) process p and p may have been influenced by decision *de*.

Definition 16. (Multi-Step MayHaveBeenInfluencedBy) An artifact a may have been influenced by a decision *de* (possibly using multiple steps) written as $a \rightarrow^* de$, if the artifact a was derived from (possibly using multiple steps) artifact a_1 and a_1 may have been influenced by decision *de*.

5. An OWL + Rules Binding of the Scientist's Intent Model

In this section we introduce a realisation of the model presented in Section 4.1 achieved by combining an OWL binding of the Open Provenance Model, an OWL ontology describing the Scientist's Intent extensions and rules based on the SWRL ontology.

5.1. An OWL Binding of the Open Provenance Model

As part of the PolicyGrid project[12] we have created a provenance framework to support evidence-based policy assessment where the focus is on how a particular piece of evidence was derived. The framework consists of a generic provenance ontology developed in OWL, which defines basic entities of OPM (such as *Artifact*, *Process*, *Agent*, *CausalRelationship*, *Role*, *Account* and *OTime*) as OWL classes. Furthermore, the framework supports additional domain-specific provenance ontologies that can be created by extending the concepts defined in the OPM ontology with domain specific classes. For example, in a "Social Simulation" domain ontology we might have a *Simulation Model* and a *Simulation Environment State* as a type of Artifact and a *Parameter Exploration* as a type of Process. To date we have developed a domain-specific provenance ontology describing aspects of "Social Simulation" and a "Generic Provenance" ontology describing generic *Artifacts* and *Processes* such as *Image*, *Questionnaire* and *Interview*.

The diagram in Figure 7 shows an extract of the ontologies defined in our provenance framework.

5.2. The Scientist's Intent Ontology

We now present the design of an ontology implementing the Scientist's Intent model as described in Section 4.1 by extending the provenance ontology outlined in Section 5.1.

We define the concept of a `Workflow Experiment` as a type of `opm:Process` designed to automate one or more `opm:Processes` defined in our provenance ontologies (e.g. `ParmeterExploration`, `DataCollection`, etc.). A `WorkflowExperiment` uses one or more `ComputationalResource` instances which represent the computational services (Grid, Web or local) associated with a workflow activity. Each `ComputationalResource` might have an associated ontology describing the resource as an entity but also describing

[12] www.policygrid.org

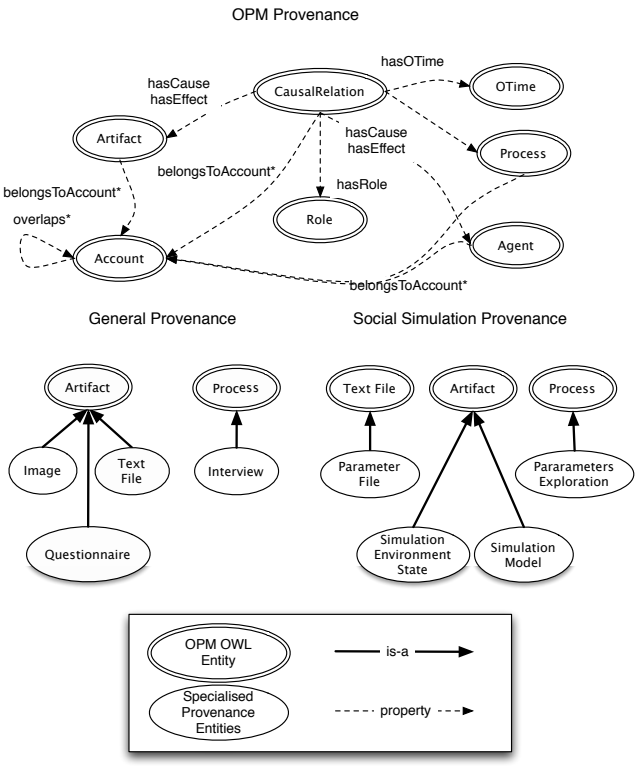


Figure 7: An Extract of our OWL-based Provenance Framework.

properties of the resource at run-time (e.g. a simulation service). A `WorkflowExperiment` inherits the characteristics of `opm:Process` and it can therefore use (`role:input`) and generate (`role:output`) `opm:Artifacts`.

A `WorkflowExperiment` can be controlled by a `WorkflowEngine` agent which characterises a specific software implementation, e.g. Kepler [11]. A `WorkflowExperiment` can also be controlled by an `SI Agent` which represents an `Agent` that is driven by scientist's intent. All `opm:Artifact` and `opm:Process` instances associated with a `WorkflowExperiment` may contain metadata produced by the workflow system. We can conveniently group such metadata in a `WorkflowState` account. A `WorkflowExperiment` may belong to more than one `WorkflowState` account capturing the temporal changes of workflow metadata. This is based on the idea of *Abstract State Spaces* [24] where a particular execution of a process denotes a sequence of state transitions $\tau = (s_0, \ldots, s_m)$ [25]. Process transitions in conjunction with time annotations (*OTime*) are required by our scientist's intent framework to reason about the `Workflow State` at runtime.

In the ontological framework presented in Section 5.1, RDF and OWL are used to express domain specific knowledge (e.g. Simulation Mode). While the OWL (version 1) language includes a rich set of class constructors, the language provided for properties is much weaker as there is no composition constructor, so it is impossible to capture relationships between a composite property and another (possibly composite) property [26]. Such a limitation makes it difficult to define the logic behind the concept of goal and constraint in an OWL (version 1) ontology as introduced by definitions 3 and 4 in Section 4.1. Goals and constraints are concerned with a desire or a restriction on a composition of properties associated with the state of a workflow. We have identified SWRL[13] (Semantic Web Rule Language) as a language for capturing rules associated with scientist's intent. SWRL enables Horn-like rules to be combined with metadata. The rules take the form of an implication between an antecedent (`ruleml:head`) and consequent (`ruleml:body`). SWRL rules can reason about OWL individuals, primarily in terms of OWL classes and properties. This formalism is particularly suitable for capturing scientist's intent as the rules can capture the logic behind goals and constraints, while the ontology and metadata about the workflow provide the 'knowledge base' upon which the rules can operate.

For example, a SWRL rule expressing that "a simulation running on a platform compatible with the IEEE 754 floating point standard will produce valid results" require us to capture the concepts of "simulation", "platform", "result" and "valid result" in OWL. The concept of simulation can be captured using an OWL class called `Simulation`, the concepts of platform, result and valid result can be expressed using OWL properties `runsOnPlatform`, `hasResult` and `hasValidResult` which are attached to `Simulation`. The rule in SWRL would then be:

$$Simulation(?x1) \wedge hasResult(?x1, ?x2) \wedge runsOnPlatform(?x1, \text{“}IEEE754\text{''}) \rightarrow hasValidResult(?x1, ?x2)$$

We choose SWRL to capture rules associated with scientist's intent because it is a declarative language not bound to a specific execution algorithm. Instead, a formal semantics is defined in order to determine when the algorithm used provides sound and complete results. This means that the scientist is able to define rules without being aware of any side effects that the rules might have as well as the algorithm that is used to execute the rules. This facilitates the use of the same rules across different scenarios. Moreover, SWRL overcomes many of the limitations of OWL in describing specific restrictions to properties. For example, OWL does not capture relationships between composite properties while SWRL does.

We introduce to our ontology the concept of `Intent` (see Figure 8) which is defined by a set of `Goal` and `Constraint` statements. An `SI Agent` might be driven by one or more `Intent` instances. `Goal` and `Constraint` are defined as subtypes of `ruleml:impl` which represent a rule axiom in SWRL. A `PreCondition` is a condition that can be satisfied on a `WorkflowState` [25] and is defines as a conjunction of `swrl:Atoms`. `Atoms` can represent an RDF resource or a built in formula. Both the `Goal` and `Constraint` class is characterised by a `Pre Condition` which contains a list of `swrl:Atom` instances.

We finally introduce the class `Decision` as a subtype of `Action` (see Figure 8) which is defined as a decision to carry on an

[13] http://www.w3.org/Submission/SWRL/

action based on a `Goal` or `Constraint` (see definition 4 and 5 in Section 4.1. An `Action` is also a conjunction of `swrl:Atom` instances which can represent an RDF resource, a built in formula or a `WorkflowAction`. Each `WorkflowEngine` agent may support zero or more `WorkflowActions`, e.g. stop workflow, pause workflow, show message.

6. Scientist's Intent Framework

To test the ideas presented in Section 4 we have implemented a framework for capturing and managing the scientist's intent associated with a workflow experiment. This framework is based on a semantically enriched workflow environment where metadata is used to annotate and describe workflow components. We have chosen to implement this framework based on the Kepler [11] workflow management system because it already provides basic semantic support. We have developed a semantic workflow architecture based on a combination of existing workflow and Semantic Web technologies (see Figure 9) in order to bridge the gap that exists between state-of the art workflow systems (e.g. Taverna, Triana, Kepler) and the type of workflow functionality required to support scientist's intent.

We begin by presenting a semantic workflow architecture based on an extension of Kepler and a number of semantic Grid and Web services. We then present an overview of the implementation of our scientist's intent framework which corresponds to the formal model and ontologies presented in Section 4.

6.1. Semantic Workflow Architecture

Our architecture is composed of three different layers. The top layer consists of components such as *Workflow Editor*, *Workflow Engine*, *Web/Grid Services* etc. which are required to design and enact scientific experiments. In our architecture, such components are described by metadata and also produce and consume metadata themselves. A the bottom layer, ontologies (e.g. OWL-S, Kepler Ontology, WSMO) are used to describe the various types of metadata produced and consumed by the other components. In the middle layer, the Semantic Bus [27] provides the means to transport metadata content between components.

The Semantic Bus in this context is a variation of the Enterprise Service Bus (ESB) ([28]) architecture used in the field of application integration. An enterprise service bus is designed to facilitate the integration of services by any application (independent of computer language or operating system) by providing an event-driven and standards-based messaging system (Bus). The key difference between a standard ESB architecture and the one we employ is that our Semantic Bus acts exclusively as a transport medium for semantic metadata between different applications and services.

The Semantic Bus provides several adapters for integrating new applications/services and ontologies into the bus (e.g. *S. I. Framework*, *S.I. Ontology*, *Simulation Ontology* described in section 6.2). The *Semantic Bus* allows different applications or services to exchange metadata by configuring incoming and outgoing endpoints.

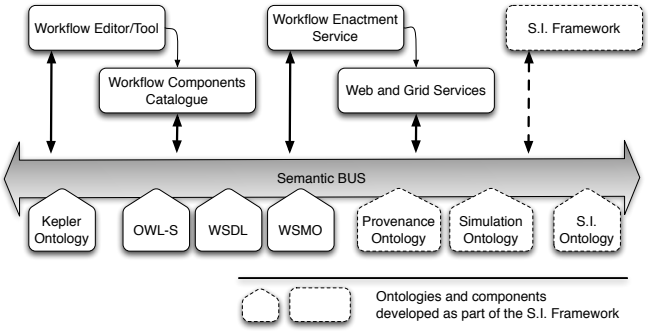


Figure 9: Semantic Workflow Infrastructure.

The top layer of our semantic workflow architecture consists of applications and services that are required to design and enact workflow experiments. This architecture is based on the core components which are common across existing workflow systems. A crucial aspect of our framework is that the workflow and its component activities (e.g. *SimulationGridTask*, *Classification/Regression*) must have supporting ontologies and should produce metadata that can be used against scientist's intent to reason about the workflow. In this section, we explain the various components of the architecture and the associated metadata support. We also discuss how adapters had to be created to allow such components to interoperate via the Semantic Bus.

In our semantic framework the *Workflow Editor* must support the creation of metadata about the workflow. Different editors allow different contextual information to be specified. Some of the most common are *title*, *author*, *date of creation*, *description*. In order to align Kepler with our provenance ontology we extended the editor to allow users to describe additional metadata e.g. `AutomateTasks` defines what high level tasks the workflow is designed to automate; `Evaluate` defines what specific hypothesis the workflow is designed to evaluate.

The *Workflow Editor* makes use of metadata available from the *Workflow Catalogue*. Such metadata can be used to automatically check if the pipeline between tasks is semantically correct (i.e. if the right type of output has been used for the right type of input). Metadata can also be used to suggest to the user suitable workflow activities based on the characteristics of their inputs and outputs. Kepler is able to import OWL ontologies and to use such ontologies to describe workflow activities. However, Kepler integrates semantic annotation directly into the workflow language (MOML) by defining properties that map to classes in an ontology. Because MOML is an XML-based language we have created an adapter to convert semantic annotations as defined in MOML to RDF statements that can be exchanged via the Semantic Bus.

The *Workflow Catalogue* contains a list of local and remote services that can be used as part of a workflow. This is a common component in workflow systems and is designed to keep an index of service locations and their characteristics, e.g. inputs, outputs, invocation details. An important function of a *Workflow Catalogue* in the context of our framework is to represent the semantic characteristics of workflow activities in a unified

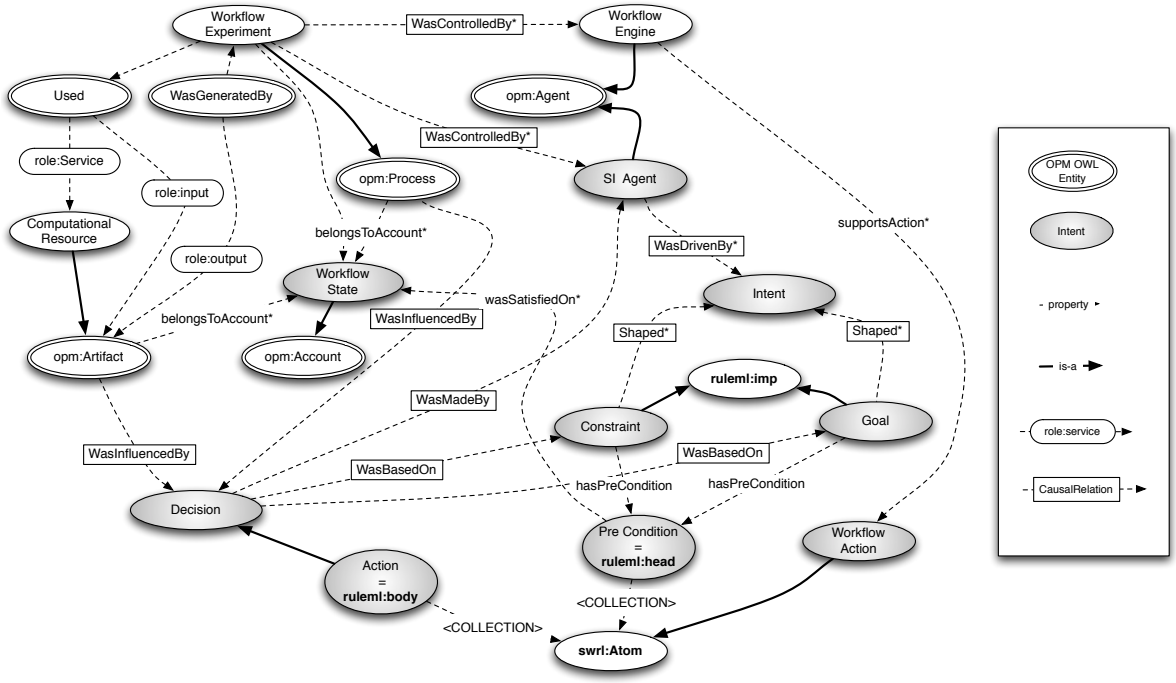


Figure 8: Intent Ontology.

way. This is complicated by the fact that the *Workflow Catalogue* can employ different mechanisms for discovering services and populating the catalogue and such mechanisms may rely on different metadata representations. In order to solve this problem the Kepler system supports annotation of workflow activities in the catalogue with "semantic type" properties. Where Kepler allows a user to manually add semantic annotations, adapters have to be created to automatically gather such information from UDDI, WSDL and OWL-S.

The recent convergence of Web and Grid technologies has facilitated the emergence of standard specifications and protocols for Web and Grid service descriptions and communication. The main specifications are the Web Services Resource Framework (WSRF) ([29]), the Web Services Description Language (WSDL) ([30]) and SOAP ([31]). We have created adapters for WSDL, WSRF, OWL-S and WSMO so that metadata about services can be shared between components of our semantic workflow architecture. Of course the usefulness of this metadata in terms of defining scientist's intent greatly depends on the standard used to describe the service. OWL-S and WSMO provide sufficient support for the semantic description of services. Another important aspect of Web and Grid services in the context of our framework is that they might produce metadata at runtime. We have implemented software components to allow the creation of such metadata at run-time by a Grid or Web service. This metadata can be made available to other components of the architecture via the Semantic Bus.

The *Workflow Engine* manages the execution of the workflow by scheduling the invocation of services and procedures based on the workflow definition. A *Workflow Engine* is tightly coupled with the *Workflow Editor* and the *Workflow Catalogue* by a *Workflow Language* so semantic support is usually limited to what the underlying language is designed to provide. In the Kepler system, the only semantic information available to the *Workflow Engine* are the semantic properties in the workflow definition itself and no additional metadata can be created. However, Kepler allows for use of customised director components that control the execution of the workflow in the workflow engine. We have created a director component (*SIDirector* described in section 6.2) that can extract metadata from the workflow and associate it with temporal information from the internal scheduler. Moreover, our director component can exchange data with other components via the *Semantic Bus* and is thus able to associate metadata from the services and the metadata from the workflow with temporal information. This is the key for creating workflow states as introduced in Section 4.

To summarise, we have created a semantic workflow architecture to provide a semantic environment within which workflow experiments can be created and executed. We have identified the following sources of metadata that can be used to support our scientist's intent framework:

- **Workflow Contextual Metadata**: metadata

about the workflow, e.g. author, date of creation, etc. Sources: Kepler Component Ontology (http://seek.ecoinformatics.org/kepler-component), Kepler Annotation Schema (http://seek.ecoinformatics.org/annotation-schema);

- **Task Metadata**: metadata about a workflow activity, e.g. task, input and output and metadata about the data generated at the end of an activity within the workflow or sub-workflow. Sources: Kepler Base Ontology (http://seek.ecoinformatics.org/ontology);

- **Service Metadata**: metadata about external services used by the workflow. Sources: OWL-S (http://www.daml.org/services/owl-s/1.2), WSDL (http://schemas.xmlsoap.org/wsdl/), WSMO: (http://www.wsmo.org/2004/d2);

- **Runtime Metadata**: metadata about the status of an activity over time, for example while the workflow is running. Sources: Simulation Ontology (http://www.policygrid.org/ontologies/simulation.owl)

6.2. Implementation of the Scientist's Intent Framework (SIF)

We have created a framework for capturing and managing scientist's intent associated with a workflow - Scientist's Intent Framework (SIF) based on the semantic workflow architecture presented in Section 6.1. Figure 10 presents the various components of such a framework and how it interacts with the Workflow Management System and Web and Grid services via the *Semantic Bus*. Each of the software components within this framework can exchange metadata.The implementation of the *Semantic Bus* consists of a set of standard APIs allowing software applications to exchange metadata.

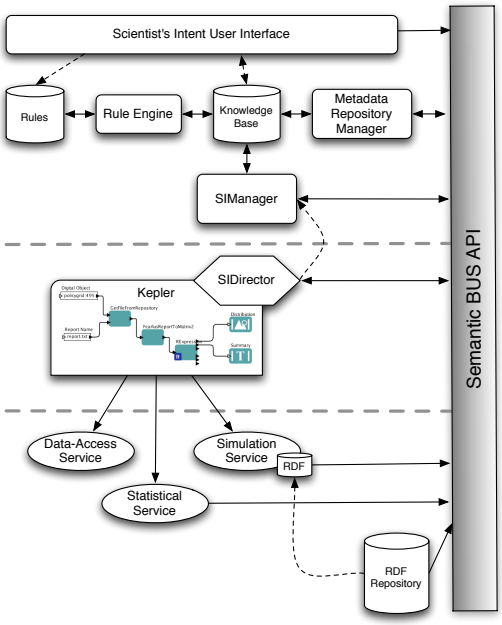


Figure 10: Scientist's Intent Management System.

At the centre of this infrastructure we have the Kepler workflow management system which allows the user to design a workflow from a catalogue of workflow activities and enact it by directly invoking local and remote services. We have implemented a number of Semantic Grid services with supporting ontologies:

- A data access service to enable access to large-scale datasets. This service is based on the existing OGSA-DAI ([32]) middleware for distributed data management with extended semantic capabilities;

- A service for statistical analysis based on R[14] which is able to provide statistical data alongside basic semantic information about the type of statistical method used;

- A number of simulation services running different versions of land-use and ecology simulation models with extensive run-time semantic metadata support.

Kepler can communicate with the rest of the framework via the *Semantic Bus* through the *SIDirector*. The *SIDirector* is an extension of a Synchronous Dataflow (SDF) Kepler Director component. The SDF Kepler Director executes a single workflow action at a time with one thread of execution. The SDF Director pre-calculates the schedule for actor execution in order to improve efficiency and reduce overhead. Moreover, a SDF director requires the same consumption and production rate of each actor e.g. an actor can only read and produce a single token of data. In our *SIDirector* implementation the SDF Kepler Director have been extended to extract metadata from the workflow actors (including input an output tokes) and to monitor and control the execution of the actors based on the assumptions resulting from the reasoning process in the Scientist's Intent Framework.

If the execution of a service produces a large amount of metadata at run-time (e.g. a simulation service), a local RDF repository for each of the service instances is created in order to avoid overloading the main RDF repository.

The core of our framework is the knowledge-base repository where metadata from the workflow and the services is translated into "facts" (represented as n-place predicates) by the *SIManager* and the *Metadata Repository Manager*. The *SIIManager* collects metadata every time it becomes available from the workflow. When some services generate a large amount of metadata, the *Metadata Repository Manager* is used to extract only the metadata required by the intent rules from the distributed RDF repositories. For example, in the FEARLUS-SPOMM Biodiversity case-study the FEARLUS model implements a mechanism to describe the status of the agents during the simulation using RDF metadata. This is facilitated by the fact that rules are expressed in SWRL and the metadata required is explicitly referenced in the rule formalism. The scientist's intent is defined by a set of rules that are stored in the *Rules Repository*. The *Rule Engine* processes such rules when new facts become available and stores the inferred facts back into

[14] http://www.r-project.org/

the knowledge-base. The same engine is also able to perform reasoning over an OWL ontology to infer additional facts.

The *SIDirector* can query the *Knowledge Base* at any time via the *SIManager* to detect if new facts have become available and if actions are required by the workflow engine. A detailed example of the use of this framework is presented in section 6.2.2.

6.2.1. Scientist's Intent Grid Service

The core of SIF is developed as a Globus Toolkit Grid service. We have used the WS-Resource[15] factory pattern so it is possible to create multiple instances of this service.

Figure 11 contains a UML class diagram describing the internal components of the Grid Service. The *SIFactory* is invoked to create one or more *SIManager* instances. An instance of *SIManager* is associated to a specific workflow experiment. Accessing individual instances of the *SIManager* is possible by the endpoint reference generated by the *SIFactory* when a new instance is created. For example, the endpoint reference is used by the *SIDirector* in Kepler for locating the appropriate *SIManager* instance to use.

An instance of the *SIManager* has an associated *KnowledgeBase* instance where facts derived from the workflow and services metadata are kept. The *SIManager* harvests RDF metadata from the workflow via the *SIDirector* component and from the services using the *MetadataRepositoryManager*. The *SIManager* has one or more *MetadataRepositoryManager* instances depending on how many services with a remote metadata repository are associated to a specific workflow. The metadata provided from the *SIDirector* and the metadata harvested from the *MetadataRepositoryManager* are stored as *WorkflowSession* instances. Such instances represent changes of the workflow and services metadata over time during the execution of the workflow.

The *SIManager* has one or more associated *RulesContainer* instances. Each *RulesContainer* represents a collection of *Rules* associated with a particular intent statement. Finally, the *SIManager* has a *RuleEngine* which is designed to run all of the *Rules* contained in the *RulesContainers* against the *KnowledgeBase* thus inferring new facts. If new facts are created a new *WorkflowSession* instance is created with the corresponding metadata.

6.2.2. Example of Usage: Baking a Cake

In this section, we present a simple scenario to illustrate the usage of the Scientist's Intent Framework in the context of the Baking a Cake example presented in Figure 5. The UML sequence diagram in Figure 12 illustrates the interaction between the software component of SIF in this example. In this scenario we will assume that we have a workflow describing the "recipe" for baking a cake and such workflow is enacted by a workflow engine.

The *SIDirector* communicates to the *SIManager* that the workflow engine is beginning to run a workflow and provides

[15] http://www.globus.org/wsrf/

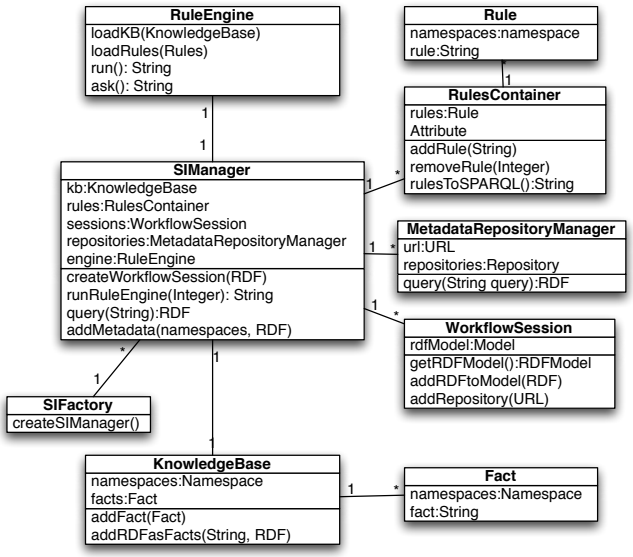


Figure 11: UML Class Diagram Describing the Scientist's Intent Grid Service.

RDF metadata describing the initial state of the workflow. This metadata is used by the *SIManager* in order to create a new *WorkflowSession* and to update the *KnowledgeBase* with the facts describing the initial state of the workflow. Once the *KnowledgeBase* is updated, the *SIManager* instantiates and runs a new *RuleEngine* which begins by loading rules from the *RulesContainer* and facts from the *KnowledgeBase*. The *RulesContainer* contains a rule describing the constraint that *"if water content of the cake mix is above 30% the mix is too runny and an additional 20g of flour needs to be added"* and is described in rule form below:

Example 1. Rule Describing a Constraint Related to the Baking a Cake Example.

```
PreCondition:
  CakeMix( ?x1 ) ∧
  hasWaterContent( ?x1, ?x2 ) ∧
  [more-than ( ?x2, 30%) = true]
Decision:
  ACTION:addFlourTo(?x1, "20g")
```

The *SIDirector* is implemented to communicate asynchronously to the *SIManager* in order to reflect the "separation of concerns" between the workflow engine executing the workflow, and the SIF framework monitoring and controlling the workflow based on intent. During the execution of the workflow, *SIDirector* detects that new metadata is available as a workflow activity (e.g. add flour, eggs and water to the mix) has just been completed. The *SIManager* therefore sends an update message to the *SIManager* containing RDF metadata describing the output of the completed activity. In this example, such metadata also contains the information that the cake mix contains 50% of water. The new metadata received by the *SIManager* is

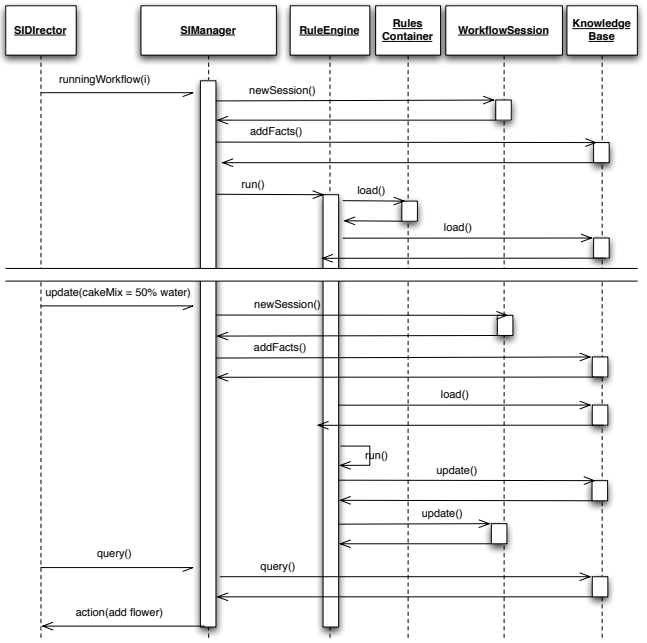


Figure 12: UML Sequence Diagram Illustrating the Baking a Cake Example.

used in order to create a new *WorkflowSession* and to update the *KnowledgeBase* with the additional facts about the workflow. The *RuleEngine* then checks the facts from the *KnowledgeBase* against the rules and any new inferred facts are stored in the *KnowledgeBase*. In this case, the fact that the cake mix contains 50% of water triggers the rule introduced above which generates the decision to add 20g of flour to the mix. This decision and related causal relationships (see Figure 5) are recorded into a new *WorkflowSession*. Throughout the process the *SIDirector* can query the *SIManager* to check if actions were originated from the *RuleEngine* and, if possible, act upon them (e.g. add 20g of flour to the mix).

6.3. Example of Usage: Workflow Control

We now present another scenario to illustrate how the Scientist's Intent Framework interacts with the Kepler director during the execution of a workflow. We will assume that a workflow experiment with associated intent rules is currently being executed, and a *SIManager* instance has already been created. The UML sequence diagram in Figure 13 illustrates the interactions between the software components of SIF. The *SIDirector* detects that new metadata is available from the workflow as a workflow activity has just been completed. The *SIDirector* therefore sends an update message (via the Semantic Bus) to the *SIManager* containing RDF statements describing the output of the completed activity. Assuming that there are some associated RDF repositories describing simulation run-time metadata, the *SIManager* generates a SPARQL query based on the rules available from the *RulesContainers*. This is done by invoking the `rulesToSPARQL` method. For example, if the *RulesContainer* contains the rule shown in Example 2, then the `rulesToSPARQL` method will return the query in Example 3 that can be used against the remote repositories to obtain the metadata necessary to check the rule.

Example 2. Rule Describing a Constraint used for Monitoring the Experiment in the Biodiversity Case-study.

```
PreCondition:
  Simulation( ?x1 ) ∧
  hasSimulationRun( ?x1, x2 ) ∧
  hasYear(?x2, ?x3 ) ∧
  hasLandManagerPopulationSize( ?x3, ?x4 ) ∧
  hasBankruptcies(?x3, ?x5 ) ∧
  [more-than( ?x5 / ?x4, 0.5 )]
Decision:
  hasInvalidRun( ?x1, ?x2) ∧
  ACTION:stop(?x1)
```

Example 3. Query for Obtaining Metadata to Validate the Constraint in the Example 2

```
DESCRIBE ?x1 ?x2 ?x3 ?x4 ?x5
WHERE
{
  ?x1 rdf:Type Simulation .
  ?x1 hasSimulationRun ?x2 .
  ?x2 hasYear ?x3 .
  ?x3 hasLandManagerPopulationSize ?x4 .
  ?x3 hasBankruptcies ?x5
}
```

The query is sent to all instances of the *MetadataRepositoryManager* via the Semantic Bus which returns a collection of RDF instances. The RDF resulting from the queries is combined with the RDF received from the *SIDirector* to form a new *WorkflowSession*. The same RDF is converted into facts by invoking the `addRDFasFacts` method of the *KnowledgeBase* class.

The *SIManager* then runs the *RuleEngine* which begins by loading rules from the *RulesContainer* and facts from the *KnowledgeBase*. The rules are checked against the facts and any new inferred facts are stored in the *KnowledgeBase* and also converted to RDF and stored in the *WorkflowSession*. The new inferred facts about the current workflow are stored in the *WorkflowSession*. Throughout the workflow execution process the *SIDirector* can query the *SIManager* via the Semantic Bus to check if actions were generated by the *RuleEngine* and have been asserted in the current *WorkflowSession*. Depending on the ability of the *SIDirector* instance to interact with the workflow engine, the *SIDirector* will act upon any detected actions by invoking the control functions in the workflow engine that coordinates suspend or resume executions.

External applications and services can access the metadata stored in the *WorkflowSession* using the Semantic Bus API. For instance, the new facts inferred as a result of the rule in Example

2 are visualised in the timeline widget of the user interface as illustrated in Figure 14. Such facts are presented in temporal order of when an action was triggered by a specific intent rule.

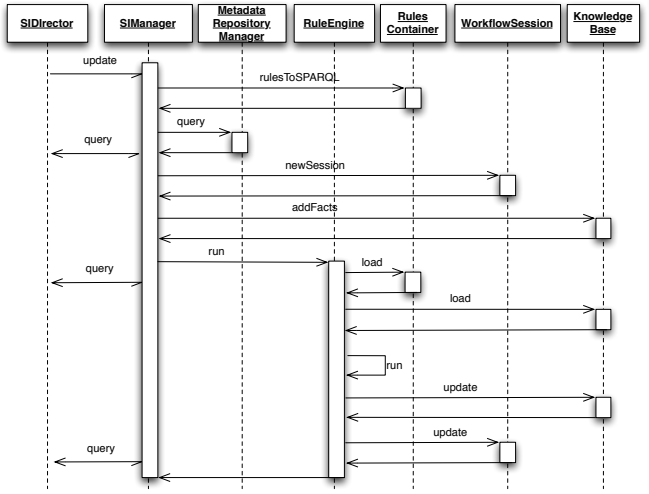


Figure 13: UML Sequence Diagram Illustrating the Workflow Control Example.

6.3.1. User Interface

We have implemented a prototype web-based user interface to create and explore scientist's intent. Figure 14 shows a screenshot of the interface in which a user is defining one of the constraints presented in the case study section.

The interface provides the user with metadata classes and properties that can be used as part of a goal or constraint (*Metadata panel*). Such metadata is based on the ontologies used to describe the workflows available to the user (*Workflow panel*). The definition of a goal or constraint is specified by the user by dragging and dropping metadata elements from the metadata panel to text boxes forming the rule statement. Built-in functions (e.g. `more-than`, `less-than`) can also be selected from the drop-down menus associated with the rule statements. Workflows on the workflow panel can be associated with one or more intent definitions and executed.

The metadata generated from the scientist's intent during the execution of a workflow is collected by the web interface via the Semantic Bus and presented back to the user in the form of a timeline using a timeline widget[16] (lower part of Figure 14). The timeline widget presents metadata elements originated by scientist's intent rules when they occurred during the execution of the workflow.

6.4. Technical and Ontological Requirements

The following are a set of requirements that are necessary for a software application or service to be used in conjunction with this software framework:

Technical Requirements

1. The application can be encapsulated in a Web or Grid service.
2. The application can run independently without user intervention at runtime.
3. OPTIONAL: The application can be controlled at runtime by another software process or service.
4. The workflow engine can be extended to communicate with the SI Framework via the Semantic Bus via Web or Grid services middleware.

Ontological Requirements

1. Standard service description metadata (WSDL or OWL-S or WSMO).
2. Metadata description of inputs and outputs.
3. If the service is "interactive", metadata description of the status of the service at runtime.
4. If the framework is associated to an existing provenance ontology, the ontology must be compatible with the OPM specifications.

7. Evaluation

In this Section, we evaluate how the implementation of the scientist's intent framework meets the aims of our research in two stages: (a) An analysis of the capability of our implementation to provide additional workflow documentation; (b) An analysis of the capability of the implementation to control and monitor the execution of a workflow. We conclude this section with a summary of a study investigating the benefits of using our scientist's intent framework in the context of social simulation.

7.1. Providing Additional Workflow Documentation

In the first stage of this evaluation we assess the capability of our implementation to provide additional workflow documentation. We demonstrate this by discussing how the sample intent queries introduced at the end of Section 3.1 can be performed using our software framework and how constraints can be used in order to enrich workflow results.

We first need to demonstrate how goals and constraints can be represented by using our framework. We therefore introduce a number of example rules using a relatively informal "human readable" syntax. In this syntax a rule takes the form:

$$PreCondition : (a_1 \wedge ... \wedge a_n), (optional) Decision : (a_1 \wedge ... \wedge a_n)$$

PreCondition and *Decision* are conjunctions of atoms ($a_1 \wedge ... \wedge a_n$) and variables are indicated using the standard convention of prefixing them with a question mark (e.g., ?x).

The rule below defines the goal of the Biodiversity case study (see Goal 1, Section 2):

Example 4. Rule Describing the Goal of the Experiment of the Biodiversity Case Study (see Goal 1, Section 2).

[16] http://simile.mit.edu/timeline/

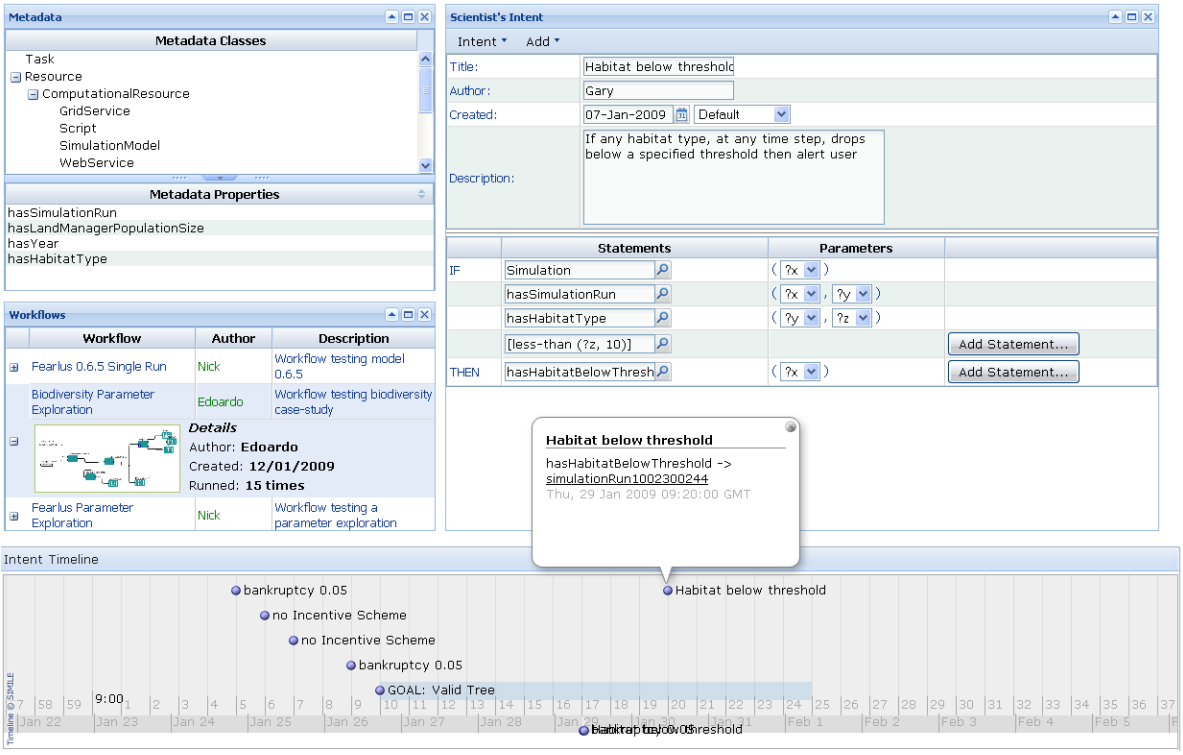


Figure 14: Intent User Interface (Biodiversity Example).

```
Pre Condition
  DataSet( ?x1 ) ∧
  ClassificationTree( ?x2 ) ∧
  hasDataSet( ?x2, ?x1 ) ∧
  hasDeviance( ?x1, ?x3 ) ∧
  hasMeanLeafNodeDeviance( ?x2, ?x4 ) ∧
  [more-than ( ?x4 / ?x3, 0.05) = true]
```

This rule states that the goal is to obtain at least one cross-validated tree that explains 5% of the deviation of the results data. This is achieved when the pre-condition occurs based on the `Artifacts` and `Processes` belonging to a `Workflow-State`. `DataSet`, and `ClassificationTree` refer to ontological classes developed as part of the PolicyGrid provenance framework (see Section 5.1), `hasDataSet`, `hasDeviance` and `hasMeanLeafNodeDeviance` are properties in those classes and `more-than` is a built-in function used to test the difference between `hasDeviance` and `hasMeanLeafNodeDeviance`.

The rule below represents the subgoal of the Biodiversity case study (see Goal 2 Section 2):

Example 5. Rule Describing a SubGoal of the Experiment in the Biodiversity Case Study (see Goal 2, Section 2).

```
Pre Condition
  ParameterPermutationsExperiment( ?x1 ) ∧
  hasSimulationRun( ?x1, ?x2 ) ∧
  hasOutcomeBasedIncentiveScheme( ?x2, ?x3 ) ∧
  hasActivityBasedIncentiveScheme( ?x2, ?x4 ) ∧
  [more-than ( ?x3 / ?x4, 0.8) = true]
```

The rule explains that a subgoal of the experiment is to obtain a similar number of valid runs between outcome-based and activity-based incentive schemes.

The rule below demonstrates how a constraint can be defined using our framework by describing Constraint 1 from Section 2.

Example 6. Rule Describing a Experimental Constraint in the Biodiversity Case Study (see Constraint 4, Section 2).

```
PreCondition:
  Simulation( ?x1 ) ∧
  hasSimulationRun( ?x1, ?x2 ) ∧
  hasLandManager( ?x2, ?x3 ) ∧
  ownsLandParcels( ?x3, ?x4 ) ∧
  [more-than ( ?x4, 50%) = true]
Decision:
  ACTION:discard(?x1)
```

The rule states that the condition: *"a simulation run ?x2 has one land manager ?x3 that owns more than half of the land"* forms the bases for the decision: *"discard the entire simulation ?x1"*. The rule can also be read in reverse: *"The decision to discard the entire simulation ?x1 was based on the condition that a simulation run ?x2 has one land manager ?x3 that owns more than half of the land"*.

7.1.1. Querying Intent

In order to leverage the expressivity of provenance queries we have combined an OWL realisation of an OPM provenance graph with an OWL realisation of intent using a custom ontology in combination with user-defined SWRL rules. This combination enables a user to issue not only provenance queries related to the primary concepts described in OPM (e.g. *Artifact*, *Process* and *Agent*) but also based on concepts associated with intent (e.g. *Decision*, *Intent*, *Goal* and constraint). Moreover, the realisation of intent using an OWL ontology linked to user-defined SWRL rules allow provenance queries to contain links to horn-like rules describing goals and constraints, e.g. pre-condition, decision/action. Using our framework it is possible to express a wider range of queries than the one allowed by the OPM OWL ontology alone. We now discuss one by one the intent queries introduced earlier in this paper.

(Q1) What was the intent of agent Ag_x ? (also related to query CS1)

In the model of scientist's intent presented in this paper we state that the intent of agent is defined by a set of goals and constraints. In our model, we represent *Intent*, *Goal* and *Constraint* and we associate them with an *Agent* in the OPM by providing two causal relationships: "was driven by" ($\langle ag_1, in_1, acc\rangle \in WasDrivenBy$), and "was based on" ($\langle in_1, gl_1, acc\rangle \in WasBasedOn$ and $\langle in_1, co_1, acc\rangle \in WasBasedOn$). Note that the edges resulting from both relationship are members of an account.

In order to answer the query: *"What was the intent of agent Ag_x?"* we need to find all the goals and constraints associated with an account as described by the following equation:

$$\begin{aligned} For\ ag_1, acc\ \ find\ all\ \ gl_1, co_1\ \ where\ \exists in_1,\\ \langle ag_1, in_1, acc\rangle \in WasDrivenBy\\ (\ \wedge \langle in_1, gl_1, acc\rangle \in Shaped\\ \vee \langle in_1, co_1, acc\rangle \in Shaped\ ) \end{aligned} \tag{1}$$

which can easily be applied to our provenance repository using the SPARQL query as demonstrated in Example 7.

Example 7. Query for Obtaining the List of Goals and Constraints Associated with an Agent in a Specific Account.

```
SELECT ?goalOrConstraint
WHERE
{
  ?agent rdf:Type opm:Agent .
  ?wdb rdf:Type si:WasDrivenBy .
  ?wdb opm:hasCause ?agent .
  ?wdb opm:hasEffect ?intent .
  ?wdb opm:belongsToAccount ?account .
  ?wdfb rdf:Type si:Shaped .
  ?wdfb opm:hasCause ?intent .
  ?wdfb opm:hasEffect ?goalOrConstraint .
  ?wdfb opm:belongsToAccount ?account .
  FILTER (?agent = <ag1> || ?account = <acc1>)
}
```

(Q2) What was the intent of agent Ag_x while controlling a process P_x? (also related to query CS1)

In order to answer this query we can use a similar approach to the one that we use in Q1 but restrict the query to only a specific process by using the relationship *WasControlledBy* between an *Agent* and a *Process*. In order to answer the query: *"What was the intent of agent Ag_x while controlling a process P_x?"* we need find all the goals and constraints associated with a process P_x, as described by the following equation:

$$\begin{aligned} For\ ag_1, p_1\ \ find\ all\ \ gl_1, co_1\ \ where\ \exists in_1,\\ \langle ag_1, in_1, acc\rangle \in WasDrivenBy\\ \wedge \langle p_1, ag_1, acc\rangle \in ControlledBy\\ (\ \wedge \langle in_1, gl_1, acc\rangle \in Shaped\\ \vee \langle in_1, co_1, acc\rangle \in Shaped\ ) \end{aligned} \tag{2}$$

which can be applied to our provenance repository using the SPARQL query as demonstrated in Example 8.

Example 8. Query for Obtaining the List of Goals and Constraints Associated with an Agent while Controlling a Specific Process.

```
SELECT ?goalOrConstraint
WHERE
{
  ?agent rdf:Type opm:Agent .
  ?crel rdf:Type opm:WasControlledBy .
  ?crel opm:hasCause ?process .
  ?crel opm:hasEffect ?agent .
  ?wdb rdf:Type si:WasDrivenBy .
  ?wdb opm:hasCause ?agent .
  ?wdb opm:hasEffect ?intent .
  ?wdfb rdf:Type si:  Shaped .
  ?wdfb opm:hasCause ?intent .
  ?wdfb opm:hasEffect ?goalOrConstraint .
  FILTER (?agent = <ag1> || ?process = <p1>)
}
```

The case-study query *"(CS1) What was the intent of the scientists while running this experiment?"* can now be answered by the solutions described for (Q1) and (Q2).

(Q3) What decisions has agent Ag_x made while controlling a process P_x? (also related to query CS4)

In order to answer this query we need find all the decisions that an agent Ag_x made by controlling a process P_x as described by the following equation:

$$\begin{aligned} For\ ag_1, p_1\ \ find\ all\ \ de_1\ \ where\ \exists de_1,\\ \wedge\langle de_1, ag_1, acc\rangle \in WasMadeBy\\ \wedge \langle p_1, ag_1, acc\rangle \in ControlledBy \end{aligned} \tag{3}$$

which can be applied to our provenance repository using the SPARQL query as demonstrated in Example 9.

Example 9. Query for Obtaining the List of Decisions made by the Agent while Controlling a Process.

```
SELECT ?constraint
WHERE
{
  ?agent rdf:Type opm:Agent .
  ?wmb rdf:Type si:WasMadeBy .
  ?wmb opm:hasCause ?decision .
  ?wmb opm:hasEffect ?agent .
  ?wib rdf:Type si:WasInfluencedBy .
  ?wib opm:hasCause ?process .
  ?wib opm:hasEffect ?decision .
  FILTER (?agent = <ag_1> || ?process = <p_1>)
}
```

(Q4) Were any constraints defined by the agent Ag_x violated? (also related to query CS3)

In order to answer the query: *"Were any constraints defined by the agent Ag_x violated?"* we need to find if any constraint defining the intent of an agent Ag_x influenced a decision as described by the following equation:

$$\begin{aligned} For\ ag_1, acc_1\ \ find\ all\ \ co_1\ \ where\ \exists de_1, \\ \langle ag_1, in_1, acc\rangle \in WasDrivenBy \\ \wedge\ \langle p_1, ag_1, acc\rangle \in ControlledBy \\ \wedge \langle in_1, co_1, acc\rangle \in Shaped \\ \wedge \langle de_1, co_1, acc\rangle \in WasBasedOn \end{aligned} \tag{4}$$

which can be applied to our provenance repository using the SPARQL query as demonstrated in Example 10.

Example 10. Query for Obtaining the List of Constraints Violated by a Specific Process.

```
SELECT ?constraint
WHERE
{
  ?agent rdf:Type opm:Agent .
  ?wdb rdf:Type si:wasDrivenBy .
  ?wdb opm:hasCause ?agent .
  ?wdb opm:hasEffect ?intent .
  ?wdb opm:belongsToAccount ?account .
  ?wdfb rdf:Type si:  Shaped .
  ?wdfb opm:hasCause ?intent .
  ?wdfb opm:hasEffect ?constraint .
  ?wfdb opm:belongsToAccount ?account .
  ?constraint rdf:Type si:Constraint .
  ?wbo rdf:Type si:wasBasedOn .
  ?wbo opm:hasCause ?decision .
  ?wbo opm:hasEffect ?constraint .
  ?wbo opm:belongsToAccount ?account .
  FILTER (?agent = <ag_1> || ?account = <acc_1>)
}
```

(Q5) Did an agent Ag_x achieve his goal/goals? (also related to query CS2)

The abstract model of intent presented in Section 4 lacks the expressive power to answer this query as there are no any causal relationships that we can use to infer or assert that a *Goal* has been achieved by an *Agent*. Asserting that a *Goal* has been achieved depends on the way that the goal is constructed (implemented) and it is beyond the scope of the Scientist's Intent abstract model to represent this. However, the realisation of the model using OWL and rules can offer support for such a query. For example, consider the goal described below (see Goal 1, Section 2) using the rule formalism introduced in our Scientist's Intent model:

Example 11. Rule Describing the Goal of the Experiment of the Biodiversity Case Study (see Goal 1, Section 2).

```
Pre Condition
  ParameterSet( ?x1 ) ∧
  DataSet( ?x2 ) ∧
  ComparisonTest( ?x3 ) ∧
  compares( ?x3, ?x1 ) ∧
  compares( ?x3, ?x2 ) ∧
  similarity( ?x3, ?x4 ) ∧
  [more-than ( ?x4, 0.95) = true]
```

This states that the goal is to obtain at least one match where the real data falls within a 95% confidence interval of the model value. This is can be checked against the provenance repository when the `Pre Condition` occurs based on the `Workflow State` using the reasoning capabilities of the Rule Engine developed as part of the Scientist's Intent Framework. The example in Figure 14 shows how such a capability is put into action via the user interface. The timeline in Figure 14 shows when the experimental goal has been achieved.

(Q6) What are the decisions made by agent Ag_x that have influenced an artifact A_x?

In order to answer this query we need find all the decisions made by an agent Ag_x that might or have influenced an artifact A_x as described by the following equation:

$$\begin{aligned} For\ ag_1, a_1\ \ find\ all\ \ de_1\ \ where\ , \\ \wedge\langle de_1, ag_1, acc\rangle \in WasMadeBy \\ (\ \wedge\langle a_1, de_1, acc\rangle \in WasInfluencedBy \\ \vee\langle a_1, de_1, acc\rangle \in MayHaveBeenInfluencedBy^*) \end{aligned} \tag{5}$$

Note that in the equation above we are using the transitive version of the *MayHaveBeenInfluencedBy* relationship in order to find all of the decisions that may have indirectly influenced the artifact involving multiple transitions. This type of query is not possible in our repository as SPARQL 1.0 does not support constraint path expressions of arbitrary length. We have therefore to limit this kind of query only to direct occurrences of the *WasInfluencedBy* relationship between an *Artifact* and a *Decision*. An example of such a query is shown in Example 12.

Example 12. Query for Obtaining the List of Decisions made by and Agent that have Influenced an Artifact.

```
SELECT ?decision
WHERE
{
  ?agent rdf:Type opm:Agent .
  ?wmb rdf:Type si:wasMadeBy .
  ?wmb opm:hasCause ?decision .
  ?wmb opm:hasEffect ?agent .
  ?owib rdf:Type si:wasInfluencedBy .
  ?owib opm:hasCause ?artifact .
  ?owib opm:hasEffect ?decision .
  OPTIONAL {
  ?crel rdf:Type opm:WasGeneratedBy .
  ?crel opm:hasCause ?artifact .
  ?crel opm:hasEffect ?process .
  ?owib rdf:Type si:wasInfluencedBy .
  ?owib opm:hasCause ?process .
  ?owib opm:hasEffect ?decision .
  } .
  ?agent opm:belongsToAccount ?account .
  ?intent si:Shaped ?constraint .
  ?constraint rdf:Type si:Constraint .
  ?decision si:wasBasedOn ?constraint .
  FILTER (?agent = <ag1> || ?account = <acc1>)
}
```

In order to answer the query (Q6), a reasoning engine based on the inference rules introduced by the Open Provenance Model and our Scientist's Intent Model has to be developed. For this reason we can only provide a partial solution to answer *"(CS4) What decisions were made while executing this experiment?"*. Developing such a reasoning engine is beyond the scope of this paper and it is currently under development by the PolicyGrid project.

7.1.2. Facilitating the Interpretation of Experimental Results

To demonstrate the capability of our implementation to provide additional workflow documentation, we introduce additional example rules based on the simulation case-study. In the Biodiversity experiment if any habitat type, at any time step, drops below a specified threshold, it is interesting to explore the simulation. The constraint below adds a new property (`hasHabitatBelowThreshold`) to the `Simulation` instance.

Example 13. Rule Describing a Constraint used for Result Enrichment in the Biodiversity Case Study (see Constraint 2, Section 2).

```
PreCondition:
  Simulation( ?x1 ) ∧
  hasSimulationRun( ?x1, ?x2 ) ∧
  hasHabitatType( ?x2, ?x3 ) ∧
  [less-than ( ?x3, 10) = true]
Decision:
  hasHabitatBelowThreshold(?x1, ?x2 )
```

Using this new property, it is possible to explore the simulation data after the workflow has been completed by following the annotations provided by the scientist's intent, e.g. `hasHabitatBelowThreshold`. The simulation instance contains a link to the repository containing the relevant simulation metadata. By exploring such metadata the scientist can gain an insight into the simulation model status and understand the mechanism(s) that triggered a particular event. For example, this new information about the simulation model can be used to define new constraints that can be used during another experiment.

Another example constraint is presented below:

Example 14. Rule Describing a Constraint used for Result Enrichment in the Biodiversity Case Study (see Constraint 3, Section 2).

```
PreCondition:
  ClassificationTree( ?x1 ) ∧
  IncentiveScheme( ?x2 ) ∧
  neg hasVariable(?x1, ?x2 ) ∧
Decision:
  hasNotIncentiveSchemeVariable( ?x1, ?x1 )
```

This informs the user when a classification/regression tree does not identify Incentive Scheme as an influential variable. In this constraint, the statement `neg hasVariable(?x1, ?x2 )` is a negation as failure based on the closed world assumption (what is not currently known to be true is false). As a consequence, if the variable `IncentiveScheme` was never used by the `ClassificationTree` such a statement is considered to be false.

7.2. Monitoring and Controlling Workflow

In the second stage of this evaluation, we assess the capability of our implementation to control and monitor the execution of a workflow. We demonstrate this by discussing how actions on the workflow execution can be triggered by intent constraints. We also discuss the performance issues related to the use of intent to monitor the execution of a workflow.

Using our framework, it is possible to control the execution of a workflow by specifying a post action from a number of options coded in an ontology, e.g. `stop workflow`, `pause workflow`, etc. As described in the usage example in section 6.2.2, the SIF is able to monitor the execution of the workflow by receiving continuous updates from the workflow engine regarding the state of the workflow over time. If a particular condition occurs that requires an action to be performed in the workflow, the SIF is able to generate action statements that can be used to control the executions of the workflow. For example, the constraint below is used to check if the simulation is running on a platform compatible with the IEEE 754 floating point standard:

Example 15. Rule Describing a Constraint on the Platform Running a Simulation (see Constraint 5, Section 2).

```
PreCondition:
  GridTask( ?x1 ) ∧
  Simulation( ?x2 ) ∧
  runsSimulation( ?x1, ?x2 ) ∧
  neg runsOnPlatform( ?x1, 'IEEE754' ) ∧
  hasResult( ?x2, ?x3)
Decision:
  hasInvalidResults( ?x2, ?x3 )
  ACTION:resubmitTask(?x1)
```

Actions based on scientist's intent (e.g. `resubmitTask(?x1)`) depend on the ability of the workflow to process events triggered by the scientist's intent framework. In our case, the extended Kepler Director component is able to understand the above action and therefore re-submits a Grid task.

In the Biodiversity experiment, a wide range of possible combinations of parameter values are explored. It is interesting here to narrow the parameter space to be searched in order to save computing resources and to gain understanding of the relative importance and major interactions between input parameters. For example, if a simulation run has a bankruptcy rate of more than 5%, ignore this parameter set. The constraint in Example 2 demonstrates how this can be achieved. The action `stop(?x1)` stops the entire simulation when one of the runs violates the pre condition.

7.2.1. Performance Issues

In a *scientist's intent* aware workflow engine, intent information can be used to reduce the time an experiment takes to complete. However, there are some performance issues to consider when using our Scientist's Intent Framework.

We introduce here a T-Complexity formula of the Biodiversity workflow experiment introduced in Section 2:

$$(per \times t_1) + (per \times files \times t_2) + (per \times t_3 \times runs) + (per \times t_4)$$

Where: per = Number of parameter permutation, t_1 = Time to generate a parameter permutation, t_2 = Time to upload a parameter file, t_3 = Time to execute a simulation run, t_4 = Time to execute a classification/regression activity, $files$ = Number of files in a parameter set, $runs$ = Runs per parameter permutation.

Activities in a workflow are commonly executed in a heterogeneous environment (e.g. a Grid cluster), the time that it takes for an activity to complete can not be reliably predicted. Therefore, we have conducted a number of workflow experiments in a controlled computing environment to test the system performance. We used a dedicated machine to calculate the average time that the activities (t_1, t_2, t_3 *and* t_4) take in the Biodiversity workflow under different scenarios:

- With or without metadata support (MD = Y, MD = N), e.g. metadata generated by the simulation service, metadata about the comparison test;
- Without the support of the Scientist's Intent Framework (IN = N);
- With full support from the Scientist's Intent Framework (IN = F), i.e. checking every change in workflow state including runtime metadata from the simulation service;
- With partial support from the Scientist's Intent Framework: checking every change in workflow state and (a) Every 4 interactions (years) of the simulation run (IN = 4Y) (b) At the beginning and end of the simulation (IN = BE);
- Dropping simulation runs as a result of actions influenced by intent: no drops (SIM DROP = N), 50% drops (SIM DROP = 50%).

Table 1 shows a summary of the data generated by the performance experiment. The first row of data on the table shows the "controlled" scenario where the workflow experiment was performed without generating metadata and without the aid of the Scientist's Intent Framework. The other rows of data show the different scenarios tested by the experiment. We notice here that in order to reduce the time it takes to run the "controlled scenario" (15,581.17 CPU/hours) it is necessary to reduce the times that intent is checked against the simulation data (IN = 4Y) and to drop 50% of the simulation runs as a result of intent actions.

Based on the data generated by this performance experiment, we can infer the following:

- With full intent support, in order to achieve a reduction on the time it takes to perform the experiment 75% of the simulation runs have to be dropped as a result of intent.
- With intent support for every 4 interactions of the simulation run, in order to achieve a reduction on the time it takes to perform the experiment, 45% of the simulation runs have to be dropped as a result of the intent.
- With intent support at the beginning and end of the simulation, in order to achieve a reduction on the time it takes to perform the experiment 39% of the simulation runs have to be dropped as as result of the intent.

7.3. Using the Scientist's Intent Framework for Social Simulation

As part of our evaluation we conducted a focus group discussion with three social simulation scientists involved in our case studies in order to discuss four key issues surrounding the use of our Scientist's Intent framework : (a) Result enrichment and annotation; (b) Reusability of experimental goals and constraints; (c) Provenance about the experiment; (d) Monitoring of an experiment.

We now present some of the most significant points to emerge from that discussion:

All participants agreed that workflow technologies can provide a structured record of the execution of a social simulation

per	t1	t2	t3	t4	files	runs	T (CPU/hr)	MD	IN	DR
5188	0.10	1.03	1.52	0.24	17	18	15,581.17	N	N	N
5188	0.10	1.03	12.44	0.24	17	18	24,079.12	Y	N	N
5188	0.10	1.03	63.00	0.24	17	18	63,424.91	Y	F	N
5188	0.10	1.03	16.94	0.24	17	18	27,581.02	Y	4Y	N
5188	0.10	1.03	13.14	0.24	17	18	24,623.86	Y	BE	N
5188	0.10	1.03	63.00	0.24	17	9	32,098.47	Y	F	50%
5188	0.10	1.03	16.94	0.24	17	9	14,176.52	Y	4Y	50%
5188	0.10	1.03	13.14	0.24	17	9	12,697.94	Y	BE	50%

Table 1: Summary of Workflow Experiments Conducted to Evaluate the System Performance.

experiment; this facilitates researchers discovering and interpreting experimental data. However, participants also agreed that the workflow execution log does not help to identify invalid or unrealistic simulation results. For instance, in an experiment involving the Biodiversity case-study, scientists found that it was very difficult to remember the exact details of the experimental parameters. One of the scientists said: *"we could have easily defined a rule saying, any set of runs in which there exists a certain proportion of the runs where the bankruptcy rate was more than 5% can be discarded"*.

The group also discussed the issue of reusability of agent-based models. If a scientist wants to replicate a simulation experiment (s)he will need to reuse the workflow designed for the experiment. However, there are a number of constraints that have to be respected in order to replicate an experiment. For instance, the SPOMM model has some general constraints, such as: *"you might drop runs where there is extinction after 50 time steps"*. A participant concluded that *"if it was standard practice to define constraints for an experiment it will be important to re-use such constraints"*. Another participant suggested that you could have a spatial metadata constraint such as: *"this model is only appropriate to the kind of social system that you find in Scotland"*.

The group suggested that there might be situations in which scientist's intent could be used to facilitate the interface between research and management. For example, in any politically contentious areas where simulation models are involved, researchers are likely to be faced with questions such as: Does the model show what it intended to show? How are the expected results obtained? Our scientist's intent framework could provide some clues to answer such questions.

The group also discussed issues about provenance suggesting that a constraint exists in the area of social simulation where a model should be based on evidence. Specifically, no data can be used in such a model that has not come from some piece of evidence. This issue has deep implications in terms documenting and assessing how the data used in the workflow has been collected. However, the group indicated that it is no longer the case that the workflow is a sufficient representation of what has been done in an experiment, the workflow plus the goals and constraints associated with such provenance are needed in order to adequately reflect the experiment.

In terms of monitoring, the group agreed that it will be ideal to be able to see how many runs are left, what has been done so far, how long it has taken, an estimate of how long it will take to complete the runs left, how much memory the runs have been using, etc. It was also agreed that scientist's intent can help as it is possible to define constraints about the service (running the model), the model, and the processes itself. It is also possible to define constraints that help to monitor and control the execution of the experiment, a participant said *"if you are running an experiment with thousands of simulations and a lot of simulations start to go wrong because of certain a condition happening, (e.g. too much memory or wrong type of parameters) it is not necessary to keep using computing resources"*.

8. Related Work

In this section, we explore the current literature on workflow technologies, identifying the limitations of current workflow languages and provenance frameworks in capturing aspects of scientist's intent.

8.1. Workflow Languages and Systems

Many of the concepts underlying today's e-Science workflow technologies originated from business workflows. These typically describe the automation of a business process, usually related to a flow of documents. A scientific workflow, on the other hand, is about the composition of structured activities (e.g. database queries, simulations, data analysis activities, etc.) that arise in scientific problem solving [11]. However, the underlying representation of the workflow remains the same (data and control flow). Workflow technologies have emerged as an alternative for constructing scientific experiments compared to ad-hoc approaches such as shell scripts [11, 9, 33]. We now present a number of workflow languages and systems relevant to our work.

The BPEL language [8], originally designed for business, has been adapted for scientific workflow use. BPEL4WS is an extension of BPEL that provides a language for the formal specification of processes by extending the Web services interaction model to enable support for business transactions. The main limitation of BPEL is that it does not support the use of semantics to describe the workflow components and their interactions but instead relies entirely on Web services described by WSDL (Web Service Description Language). This type of language in not suitable for documenting aspects of scientist's intent as we need rich metadata support for the workflow to describe not only service related information (e.g. `platform`, `inputs`

and `outputs`) but also high level concepts (e.g. `Simulation`, `Landmanager` and `LandParcel`).

XScufl [9] is a simple workflow orchestration language for Web services which can handle WSDL based web service invocation. The main difference from BPEL is that XScufl is their control logic, were XScufl is only capable of sequential invocation of workflow activities, BPEL implements both sequential and event triggered control logics. Taverna [9], is a tool developed by the myGrid[17] project to support 'in silico' experimentation in biology, which interacts with arbitrary services that can be wrapped around Web services. The semantic support in Taverna allows the description of workflow activities but is limited to facilitating the discovery of suitable services during the design of a workflow via an ontology-driven search facility.

MoML [7] is a language for building models as clustered graphs of entities with inputs and outputs. Kepler [11] is a workflow tool based on the MoML language where Web and Grid services, Globus Grid jobs, and GridFTP can be used as components in the workflow. Central to Kepler is the use of *Directors* which define execution models and monitor the execution of the workflow. Kepler also supports the use of ontologies to describe actors' inputs and outputs, enabling it to support automatic discovery of services and facilitate the composition of workflows. Like other workflow tools, Kepler does not allow the use of metadata at runtime. However, we believe that Kepler is well suited for documenting and managing scientist's intent as the *Director* component and the integration of ontologies with workflow activities provide an ideal framework for our work.

Triana [33] is a workflow environment that focuses on integrating different types of middleware such as peer-to-peer and Grid. The approach in Triana is similar to the one in Kepler in that the workflow is constructed from "actors" that encapsulate local or remote processes. This makes Triana another potential environment for the documentation of scientist's intent, providing access not only to Grid or local services but also to peer-to-peer services.

Pegasus (Planning for Execution in Grids) [34] was developed as part of the GriPhyN[18] project. Pegasus is a system designed to map and execute complex workflows using Grid-based middleware. Pegasus is able to map an abstract workflow to a concrete workflow and submit it to Condor's DAGMan[19] for execution. The abstract workflow describes processes and data by their logical names while the concrete workflow contains the location of the data and the execution platforms. In the context of our scientist's intent framework it will be interesting to explore the idea of abstract workflows. Abstract workflows may give us the advantage to hide the operationalisation of the workflow while allowing intent to be defined as a higher level element of the workflow.

As part as the Pegasus project, Kim et al. [35] present some interesting work on generating and validating large workflows by reasoning on the semantic representation of workflow. Their approach relies on semantic descriptions of workflow templates and workflow instances. This description includes requirements, constraints and data products which are represented in ontologies. This information is used to support the validation of the workflow but also to incrementally generate workflow instances. Although in our research we are not focusing on assisted workflow composition, we do share the same interest in the benefit of enhanced semantics in workflow representation. While both our requirements rely on the existence of higher-level workflow metadata, we are taking a more user-centred approach by capturing higher level methodological information related to scientist's intent, e.g. `valid simulation result`, `land manager`, etc.

8.2. Provenance of Scientific Workflows

Documenting the provenance of a scientific workflow has been identified as an essential step to support reproducibility, scientific discovery and result interpretation [13, 12]. In general terms, provenance (also referred to lineage or audit trail) captures the derivation history of a data product, including the original data sources, intermediate data products, and the steps that were applied to produce the data product. Heinis et al. [36] argues that "without lineage information, a data set is often useless from a scientific point of view". In the context of workflow experiments lineage information is used to support scientists in different ways [37]: (a) to explore data sources (and their derivation) used and produced by the workflow; (b) to verify the data produced by the workflow for correctness; (c) to allow invalid data sources to be corrected and re-run only the activities that have been affected by the invalid data.

Clifford et al. [38] introduce three distinct form of provenance: *retrospective provenance* which captures information about the steps that were executed in order to derive some data, *prospective provenance* which captures the specification of the steps that need to be followed to obtain some data and *user-defined provenance* which represents provenance information provided by the user. A workflow can be used to trace the lineage of a data set as it already describes what tasks were performed in order to produce a data set. Existing workflow systems (e.g Kepler, Taverna and Triana) capture such information implicitly in an event log or can be easily instrumented to capture data provenance. However, the basic lineage information provided by the workflow systems are not enough because scientists are interested not only in queries related to basic lineage information such as "What algorithms were used to derive this data set?", "Which data sets have been produced with this algorithm?", "What data sets have been derived from this data set?" [36] but also to the kind of provenance queries presented in this paper. Despite the developments of user-oriented provenance frameworks [39, 40], little support exists in current frameworks to allow *"scientifically meaningful"* provenance queries, in particular when the information available about a workflow goes beyond data lineage (e.g. by also describing aspects of scientist's intent).

To date several approaches have been discussed to capture and model provenance [40, 41, 42, 43, 38, 44]. Some of these approaches [40, 43] combine together workflow systems with

[17] www.mygrid.org.uk

[18] http://www.usatlas.bnl.gov/computing/grid/griphyn/

[19] http://www.cs.wisc.edu/condor/

Semantic Web technologies (e.g ontologies, reasoning, and rules). The Semantic Web is an ideal environment for representing provenance information as the semantics of RDF and OWL allow us to define terms such that a reasoner can infer connections among different aspects of a workflow (e.g. process, data, services). One of the most relevant provenance models discussed in the literature is the Open Provenance Model (OPM) [17], an abstract model developed to address issues in exchanging provenance information in workflow-driven science. The Open Provenance Model is designed to make assertions about causation between processes and data and between processes and agents and also defines inference rules that can be applied to a provenance graph. However, while OPM defines the concept of an *agent* as an entity which is in charge of controlling a process, it does not acknowledge that agents might be autonomous (e.g. a scientist) able to make their own decisions driven by intent (goals and constraints). Relevant to this is the model of provenance in autonomous systems presented by [45]. Miles argues that *"... the intent behind any process that takes place is not fixed in the original design of the application and so cannot be understood solely by examining that design..."*. To overcome this problem the model presented by Miles combines a description of goal-oriented aspects of agency with existing provenance frameworks in service-oriented architectures. While such a model allows the goals of autonomous agents to be made explicit in the process documentation it does not take into account other aspects related to intent such as constraints and decisions. The work presented by Miles also does not explore the potential of using intent information to control, monitor and annotate a process.

8.3. Belief, Desire and Intention

Research related to the role of intent in plans, actions, rationality and intelligence has been the subject of study for many years [46]. Bratman [47] in his work on rational agents takes the view that intention is the attitude that shapes future planning. He also argues that much of our understanding of ourselves and others is based on a framework where intentions are used to characterise people's actions and their minds. The importance of intentions has been recognised in the field of intelligent agents in developing agent theories, languages and architectures. The most relevant work in this field is the model of Belief, Desire and Intent (BDI) developed by Rao and Georgeff [48]. Rao and Georgeff explain that beliefs represent the 'informative component' of a system state (i.e. the current knowledge about the world). Desires represent the objectives which the agent is trying to accomplish. Intentions are the chosen course of action in order to achieve the agent's objectives, and are generally expressed as plans and post-conditions. Agents may have multiple desires and multiple intentions running concurrently at any given time.

The rationale behind the *scientist's intent* discussed in this paper is closely related to the BDI model introduced by Rao and Georgeff. Belief is what the scientists know about the experiment (e.g. the state of the workflow during execution). Desire is what the scientist is aiming to achieve with the experiment (goals) without violating certain constraints. The plan of action for performing the experiment is described by the workflow created by the scientist. However, one must be aware that plans sometimes do not behave as intended. Intent represent the set of actions that are required to change the initial plan (workflow) in order to achieve the scientist's goals and to respect the scientist's constraints.

8.4. Scientific Applications of Workflow

The myExperiment project [49] is a social web site for scientists based on the Web 2.0 approach. myExperiment aims to deliver a social web site where scientists can safely publish and share their scientific workflows and related artifacts. The site also provides support for credit, attributions, licensing, and fine control over privacy to meet the requirements of their research users. myExperiment is also designed to share and discuss workflows and their related scientific objects, most notably provenance logs. It would be interesting to explore the effect of sharing scientist's intent information in a social networking environment like myExperiment.

Goderis et al. [50] argues that scientific research has an important social element as scientists share publications and experimental workflows with each other and it is therefore important that scientific workflows can be properly reproduced and interpreted. For this reason, myExperiment is designed to support workflow discovery, re-use and re-purposing. Goderis et al. present some interesting queries related to the re-use of workflows, for example: "Find a workflow able to replace my faulty workflow fragment" and "Find a workflow that extends my current annotation pipeline with a visualisation step". We argue that scientist's intent information could be used in this context to enable queries such as: "Find an alternative workflow to satisfy my goal", "Find a workflow able to replace the fragment of my workflow that violates my constraints". Goderis et al. also present many queries related to workflow retrieval (navigation and search), most notably: "Which workflow has used this data as input?", "Which workflows has successfully used this service?" and "People who used this workflow also used that workflow". Scientist's intent information is not currently represented in myExperiment and could potentially allow the extension of the range of retrieval queries currently possible, for example: "Which workflow is designed to achieve this goal?", "Which workflow has successfully achieved this goal?" or "Which workflow has violated these constraints?".

The SEEK project [51] aims to support acquisition, integration and analysis of ecological and biodiversity data. The Semantic Mediation System (SMS) is one of the components of the SEEK architecture and is designed to support scientists' workflow modeling and design processes. In particular, this component exploits domain ontologies to facilitate the discovery of data sets and components, binding of data sets to components, and smart linking of components to each other as part of the overall design process. The Kepler workflow tools in the SEEK architecture leverage the Semantic Mediation System by providing knowledge-based data integration and workflow composition services, as well as basic services used in workflow modeling, such as ensuring that workflows are "semantically" type-safe and component and data discovery via

concept-based searching. In such an environment information about scientist's intent could also be used to support the workflow design process by allowing: (a) the discovery of new components based on intent (e.g. could meet the scientist's goal); (b) the discovery of existing workflows that meet the scientist's goals; (c) to check if any components of the workflow violate the scientist's constraints.

9. Conclusions & Future Work

In this paper we have investigated the idea that augmenting scientific workflow with a machine-processable representation of a scientist's intent provides benefits in terms of improved workflow documentation, monitoring and control. From our evaluation we can conclude that a representation of scientist's intent can reduce human effort in inspecting workflow documentation and can provide better management of workflow execution. As part of our investigation we have presented the formal representation of OPM and discussed the challenges that we faced in order to capture scientist's intent using the existing OPM provenance representation. We introduced an extended version of OPM capturing aspects of scientist's intent. We described an OWL binding of our scientist's intent model which combines an existing OWL realisation of OPM with SWRL-based rules and a scientist's intent ontology. We presented a software framework for capturing, managing and reasoning about the scientist's intent associated with a workflow experiment. This framework can be used across different workflow engine implementations. In this section we discuss limitations of our framework and how these can be addressed in future work.

9.1. Discussion

SWRL and OWL provide a useful mechanism to express most of the concepts defined by our scientist's intent abstract model. Our evaluation showed that most of the sample intent queries introduced in this paper can be answered using our framework with some exceptions: In order to answer the query: "*(Q6) What are the decisions made by agent Ag_x that have influenced an artifact A_x?*", we need to use the transitive version of the *MayHaveBeenWasInfluencedBy* relationship in order to find all of the decisions that may have indirectly influenced the artifact involving multiple transitions. This type of query is not possible with our framework as SPARQL 1.0 does not support nested queries or recursion [52]. In our future work, it will be interesting to develop a reasoning engine capable of such queries.

Our framework was built based on the assumption that the workflow system: (a) allow to control and monitor the execution of a workflow; (b) allow the execution of Web or Grid services as workflow activities; (c) provide rich metadata support describing the services invoked by the workflow. Based on these assumptions, the Kepler workflow environment was ideally suited for our framework as it implements the concept of a Director to control the execution of a workflow (including Web and Grid services) and uses OWL ontologies to support semantic annotation of dataset schemas, activities and their corresponding input and outputs, to provide classification and browsing of workflow activities, to check if the workflow is semantically consistent and to search for contextually relevant activities during workflow design.

As described in this paper, our framework allows the user to define goals associated with a workflow experiment. The mechanism that reasons about goals is the most basic aspect of our implementation. However, most of the available workflow engines cannot be made fully compatible with scientist's intent without a major reimplementation, and the concept of `Action` is required to overcome such limitations by providing additional metadata about the workflow state when goals are achieved and constraints are satisfied. As such, workflow engines themselves are possibly the most obvious candidate for future work and improvement. In a *scientist's intent* aware workflow engine, the planning and scheduling of the workflow execution can be optimised based on goals and constraints. In the WSMO ontology, goals are defined as the objectives that a client may have when consulting a service. Such a definition can then be used to identify the services required to achieve a specific goal. Our definition of a goal could potentially be utilised with an implementation of WSMO (such as WSMX [53]) to overcome the limitation of current workflow execution engines. We have created an engine to reason about goals and constraints associated with a workflow experiment and we would like to expand the query capability of our framework by implementing a reasoning engine for the inference rules defined by the Open Provenance Model and our Scientist's Intent Model. Our scientist's intent framework utilises SPARQL 1.0 in order to represent goals and constraints. However, SPARQL 1.0 has several limitations, e.g. lack of support for aggregation functions, sub-queries and expressions. For provenance, the most serious problem is that SPARQL 1.0 does not support path variables, constraints on path expressions, or path expressions of arbitrary length.

From a user's perspective, creating and utilising metadata is a non-trivial task: the use of a rule language to capture scientist's intent introduces additional challenges in this regard. We have addressed these issues by creating a web-based tool to compose scientist's intent rules from available metadata to associate workflows with intent and to visualise intent information. Although we are using a timeline widget to present the intent information back to the user, there are still challenges associated with metadata browsing. [54] describe a tool that provides access to RDF metadata (create, browse and query) using natural language. The tool can operate with different underlying ontologies and in the future we are planning to explore whether it could be extended to support scientist's intent metadata.

9.2. Future Work

As mentioned earlier in this paper, our framework did not implement the transitive version of some of the relationships introduced in our model. In our future work, it will be interesting to develop a reasoning engine with such capabilities. This could be accomplished by using a logic programming language such as Prolog or the iTQL query language introduced by [55].

We have also mentioned that our framework makes a limited use of goals by annotating workflow results so that when a

goal has been achieved the event is recorded and displayed as part of the workflow results. Making workflow engines fully compatible with scientist's intent is possibly the most obvious candidate for future work and improvement. The major challenge related to this will be instrumenting the workflow engine to make independent execution decisions based on intent.

Since our model of intent was developed as an extension of OPM, the concept of an *OPM profile* has been created. An *OPM profile* is intended to allow the definition of a specialisation of an OPM-based model while maintaining the compatibility with the semantics described in the OPM core specification. A valid *OPM profile* implies that all the semantics and inferences described in the core specification remain unchanged. Although at first glance our model of intent seems compatible with the *OPM profile* specifications and does not represent a completely new semantics, we are still planning to investigate this issue in order to obtain a valid *OPM profile* that can be shared with the rest of the research community.

The *Fourth Provenance Challenges Scoping Workshop*[20] held in June 2010 was seeking broad end-to-end scenarios to demonstrate how OPM can be used as an interoperable provenance technology. There were a number of patterns emerging from the scenarios proposed including "user decision points" and "why the user performed a decision" both of which were inspired by the work presented in this paper. We have committed to participate in the development of solutions for these challenges and the scientist's intent model will be used and expanded in the future to provide a solution for some of the challenges.

The PolicyGrid project, a collaboration between computer scientists and social scientists, is exploring how novel e-Science technologies can be used to support inter-disciplinary research activities, in particular, the provision of support for tools for evidence-based policy research. Evidence is used at various stages of policy making, from the design of new policies to the evaluation and review of existing policies. An evidence base supports transparency and accountability in the policy decision-making process. There is considerable potential for technologies to support such activities by providing tools to assist the collaboration and interaction between researchers using the Web. PolicyGrid is studying the possible goal setting and methodological constraints imposed by shared (project) goals in the context of interdisciplinary research projects. This includes identifying examples of such goals and an examination into how they affect (constrain) activities across the disciplinary groups within each community. The project is also exploring how shared goals should be represented and the relationship between high level project goals and constraints on discipline-specific sub-tasks. The intent approach taken in this paper will be adapted for this context.

In conclusion, we aim to provide a closer connection between experimental workflows and the goals and constraints of the researcher, thus making experiments more transparent. While scientist's intent provides additional metadata information about workflow results and provenance, we have demonstrated that its use can also facilitate improved management of workflow monitoring and execution. In addition, scientist's intent provides more information about the *why* context than traditional provenance frameworks. However, much more work is needed if we are to truly capture the intent of the scientist; the framework described here is an important step towards that ultimate goal.

[20] http://twiki.ipaw.info/bin/view/Challenge/FourthProvenanceChallenge

References

[1] N. Hara, P. Solomon, S.-L. Kim, D. H. Sonnenwald, An emerging view of scientific collaboration: Scientists' perspectives on collaboration and factors that impact collaboration, Journal of the American Society for Information Science and Technology 54 (10) (2003) 952–965. doi:10.1002/asi.10291.

[2] T. Hey, A. E. Trefethen, Cyberinfrastructure for e-science, Science 308(5723) (2005) 817–821.

[3] I. Foster, C. Kesselman, J. Nick, S. Tuecke, Grid services for distributed system integration, Morgan- Kaufmann, 2002.

[4] D. De Roure, N. Jennings, N. Shadbolt, The semantic grid: a future e-science infrastructure, Grid Computing: Making the Global Infrastructure a Reality (2003) 437–470.

[5] T. Berners-Lee, J. Hendler, O. Lassila, The semantic web, Scientific American (284) (2001) 28–37.

[6] D. Pennington, Supporting large-scale science with workflows, in: Proceedings of the 2nd workshop on Workflows in support of large-scale science, High Performance Distributed Computing, 2007, pp. 45–52.

[7] A. Lee, S. Neuendorffer, Moml — a modeling markup language in xml — version 0.4, Technical report, University of California at Berkeley (2000).

[8] T. Andrews, Business process execution for web services, version 1.1, ftp://www6.software.ibm.com/software/developer/library/ws-bpel.pdf (2003).

[9] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. Carver, K. Glover, M. Pocock, A. Wipat, P. Li, Taverna: a tool for the composition and enactment of bioinformatics workflows, Bioinformatics Journal 20(17) (2004) 3045–3054.

[10] I. Taylor, M. Shields, I. Wang, A. Harrison, Visual grid workflow in triana, Journal of Grid Computing 3 (3) (2005) 153–169. doi:10.1007/s10723-005-9007-3.

[11] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jeager, M. Jones, E. Lee, J. Tao, Scientific workflow management and the kepler system, Concurrency and Computation: Practice and Experience 18 (2005) 1039 – 1065.
URL `http://doi.wiley.com/10.1002/cpe.994`

[12] R. Bose, J. Frew, Lineage retrieval for scientific data processing: a survey, ACM Comput. Surv. 37 (1) (2005) 1–28. doi:http://doi.acm.org/10.1145/1057977.1057978.

[13] Y. L. Simmhan, B. Plale, D. Gannon, A survey of data provenance in e-science, SIGMOD Rec. 34 (3) (2005) 31–36. doi:http://doi.acm.org/10.1145/1084805.1084812.

[14] P. Groth, S. Jiang, S. Miles, S. Munroe, V. Tan, S. Tsasakou, L. Moreau, An architecture for provenance systems, Tech. rep., ECS, University of Southampton (2006).

[15] C. Goble, Position statement: Musings on provenance, workflow and (semantic web) annotation for bioinformatics, Workshop on Data Derivation and Provenance, Chicago (2002).

[16] J. A. Zachman, A framework for information systems architecture, IBM Systems Journal 26 (3) (1987) 276–292.

[17] 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. V. den Bussche, The open provenance model core specification (v1.1), Future Generation Computer Systems.
URL `http://eprints.ecs.soton.ac.uk/21449/`

[18] G. Polhill, E. Pignotti, N. Gotts, P. Edwards, A. Preece, An ontology for experimentation with a social simulation of land use change, in: Representing Social Reality, (ed. Klaus G. Troitzsch). European Simulation Association, 3rd Annual Conference, Koblenz, Germany 5-9 September 2005, 2005, pp. 324–330.

[19] J. G. Polhill, N. M. Gotts, A. N. R. Law, Imitative versus nonimitative strategies in a land use simulation, Cybernetics and Systems 32 ((1-2)) (2001) 285–307.

[20] G. Polhill, G. Gimona, N. Gotts, Analysis of incentive schemes for biodiversity using a coupled agent-based model of land use change and species metacommunity model, in: Modelling for Environment's Sake. International Congress on Environmental Modelling and Software, Ottawa, Ontario, Canada, 5th-8th July 2010, 2010.

[21] R. E. Mcgrath, J. Futrelle, Reasoning about provenance with owl and swrl rules, Presented at AAAI 2008 Spring Symposium, "AI Meets Business Rules and Process Management". Stanford, CA. March 24-26, 2008. (2008).

[22] J. D. Myers, R. E. McGrath, Cyberenvironments: adaptive middleware for scientific cyberinfrastructure, in: ARM '07: Proceedings of the 6th international workshop on Adaptive and reflective middleware, ACM, New York, NY, USA, 2007, pp. 1–3. doi:http://doi.acm.org/10.1145/1376780.1376788.

[23] K. V. Hindriks, F. S. de Boer, W. V. D. Hoek, J. jules Ch. Meyer, Agent programming with declarative goals, in: Intelligent Agents VII. Agent Theories Architectures and Languages, 7th International Workshop, ATAL 2000. Volume 1986 of LNCS, Springer, 2000, pp. 228–243.

[24] U. Keller, H. Lausen, M. Stollberg, On the semantics of functional descriptions of web services, in: Proceedings of the 3rd European Semantic Web Conference (ESWC 2006), Montenegro, 2006, pp. 605–619.

[25] M. Stollberg, M. Hepp, A refined goal model for semantic web services, in: Proceedings of the Second International Conference ion Internet and Web Applications and Services. ICIW apps; 07, 2007, pp. 17–23.

[26] I. Horrocks, P. F. Patel-Schneider, A proposal for an owl rules language, in: WWW '04: Proceedings of the 13th international conference on World Wide Web, ACM, New York, NY, USA, 2004, pp. 723–731. doi:http://doi.acm.org/10.1145/988672.988771.

[27] K. Verma, A. Kass, Model-assisted software development: Using a 'semantic bus' to automate steps in the software development process, (Accepted) Semantic Web – Interoperability, Usability, Applicability.

[28] D. A. Chappell, Enterprise Service Bus, O'Reilly Media, 2004.

[29] T. Banks, Web services resource framework (wsrf), Tech. rep., Primer v.1.2. OASIS, Committee Draft 02, 23 May 2006 (2006).

[30] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana, Web services description language (wsdl) 1.1, Tech. rep., W3C (2001).

[31] M. Gudgin, M. Hadley, N. Mendelsohn, J. J. Moreau, H. F. Nielsen, Soap specifications, Tech. rep., W3C (2003).

[32] M. Antonioletti, N. C. Hong, A. Hume, M. Jackson, A. Krause, C. Palansuriya, T. Sugden, M. Westhead, Experiences of designing and implementing grid database services in the ogsa-dai project, in: Global Grid Forum Workshop on Designing and Building Grid Services, October 2003, 2003.

[33] I. Taylor, M. Shields, I. Wang, A. Harrison, The Triana Workflow Environment: Architecture and Applications, in: I. Taylor, E. Deelman, D. Gannon, M. Shields (Eds.), Workflows for e-Science, Springer, New York, Secaucus, NJ, USA, 2007, pp. 320–339.

[34] E. Deelman, S. Callaghan, E. Field, H. Francoeur, R. Graves, N. Gupta, V. Gupta, T. H. Jordan, C. Kesselman, P. Maechling, J. Mehringer, G. Mehta, D. Okaya, K. Vahi, L. Zhao, Managing large-scale workflow execution from resource provisioning to provenance tracking: The cybershake example, in: E-SCIENCE '06: Proceedings of the Second IEEE International Conference on e-Science and Grid Computing, IEEE Computer Society, Washington, DC, USA, 2006, p. 14. doi:http://dx.doi.org/10.1109/E-SCIENCE.2006.99.

[35] J. Kim, E. Deelman, Y. Gil, G. Mehta, V. Ratnakar, Provenance trails in the wings-pegasus system, Concurr. Comput. : Pract. Exper. 20 (2008) 587–597. doi:10.1002/cpe.v20:5.
URL http://portal.acm.org/citation.cfm?id=1350745.1350748

[36] T. Heinis, G. Alonso, Efficient lineage tracking for scientific workflows, in: SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data, ACM, New York, NY, USA, 2008, pp. 1007–1018. doi:http://doi.acm.org/10.1145/1376616.1376716.

[37] U. Park, J. Heidemann, Provenance in sensornet republishing, in: Proceedings of the 2nd International Provenance and Annotation Worksho, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 280–292.

[38] B. Clifford, I. Foster, J.-S. Voeckler, M. Wilde, Y. Zhao, Tracking provenance in a virtual data grid, Concurrency and Computation: Practice and Experience 20 (5) (2008) 565–575. doi:http://dx.doi.org/10.1002/cpe.v20:5.

[39] S. Bowers, T. M. McPhillips, B. Ludäscher, S. Cohen, S. B. Davidson, A model for user-oriented data provenance in pipelined scientific workflows, in: IPAW, Vol. 4145, 2006, pp. 133–147.

[40] J. Zhao, C. Goble, R. Stevens, D. Turi, Mining taverna's semantic web of provenance, Concurrency and Computation: Practice and Experience 20 (5) (2008) 463–472. doi:http://dx.doi.org/10.1002/cpe.v20:5.

[41] J. Kim, E. Deelman, Y. Gil, G. Mehta, V. Ratnakar, Provenance trails in the wings-pegasus system, Concurrency and Computation: Practice and Experience 20 (5) (2008) 587–597. doi:http://dx.doi.org/10.1002/cpe.v20:5.

[42] S. Miles, P. Groth, S. Munroe, S. Jiang, T. Assandri, L. Moreau, Extracting causal graphs from an open provenance data model, Concurr. Comput. : Pract. Exper. 20 (5) (2008) 577–586. doi:http://dx.doi.org/10.1002/cpe.v20:5.

[43] J. Golbeck, J. Hendler, A semantic web approach to the provenance challenge, Concurrency and Computation: Practice and Experience 20 (5) (2008) 431–439. doi:http://dx.doi.org/10.1002/cpe.v20:5.

[44] I. Altintas, O. Barney, E. Jaeger-Frank, Provenance collection support in the kepler scientific workflow system, Provenance and Annotation of Data (2006) 118–132doi:10.1007/11890850_14.

[45] S. Miles, S. Munroe, M. Luck, L. Moreau, Modelling the provenance of data in autonomous systems, in: AAMAS '07: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems, ACM, New York, NY, USA, 2007, pp. 1–8. doi:http://doi.acm.org/10.1145/1329125.1329185.

[46] J. R. Searle, The intentionality of intention and action, Cognitive Science 1 (4) (1980) 27–70.

[47] M. Bratman, Intentions, Plans, and Practical Reason, Harvard University Press, 1987.

[48] A. S. Rao, M. P. Georgeff, Bdi-agents: from theory to practice, in: Proceedings of the First Intl. Conference on Multiagent Systems, San Francisco,, 1995.

[49] C. A. Goble, D. C. De Roure, myexperiment: Social networking for workflow-using e-scientists, in: WORKS '07: Proceedings of the 2nd workshop on Workflows in support of large-scale science, ACM, New York, NY, USA, 2007, pp. 1–2. doi:10.1145/1273360.1273361.
URL http://dx.doi.org/10.1145/1273360.1273361

[50] A. Goderis, D. D. Roure, C. Goble, J. Bhagat, D. Cruickshank, P. Fisher, D. Michaelides, F. Tanoh, Discovering scientific workflows: The myexperiment benchmarks, Tech. rep., School of Electronics and Computer Science, University of Southampton (April 2008).
URL http://eprints.ecs.soton.ac.uk/15662/

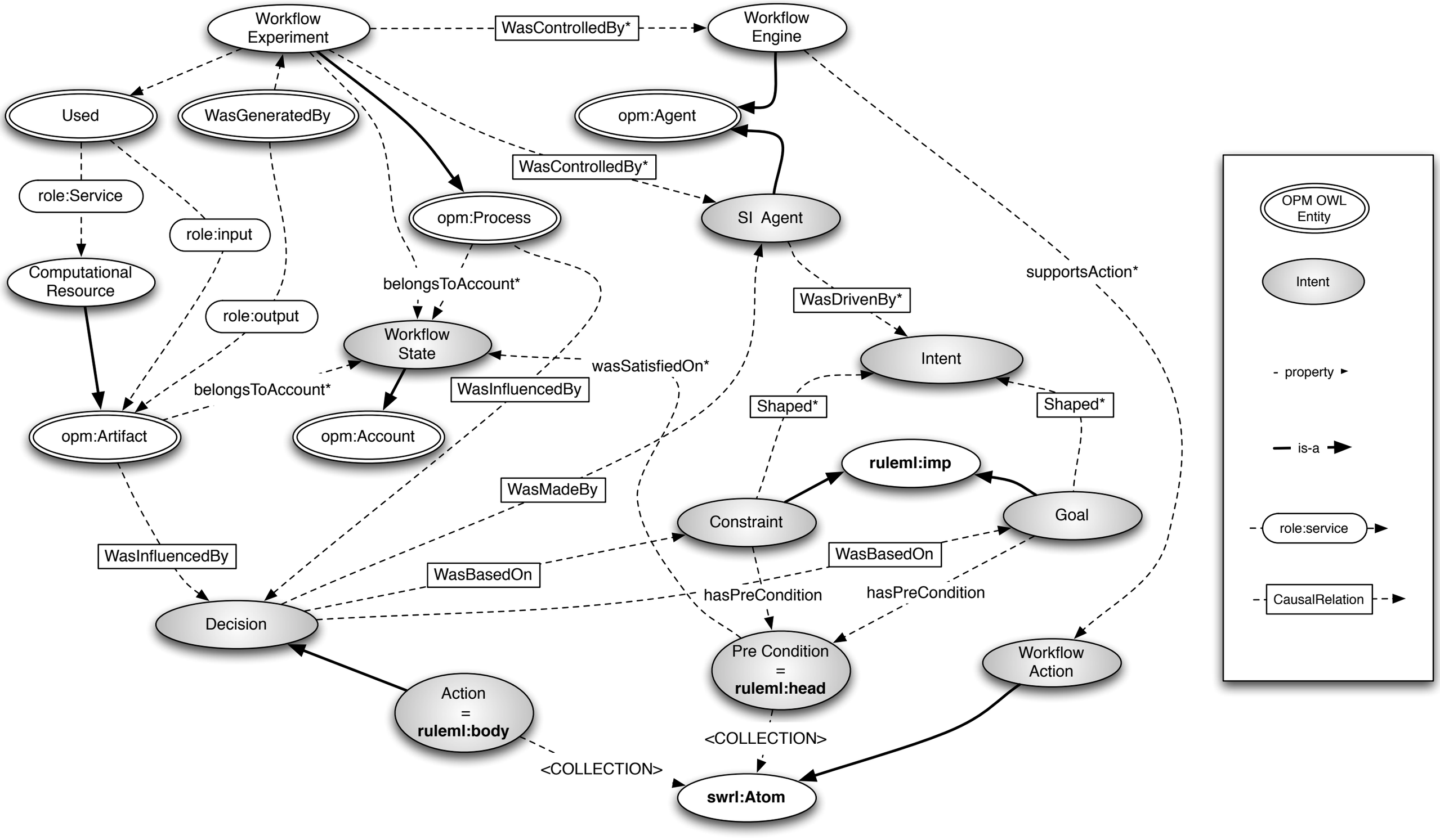
[51] W. Michener, J. Beach, S. Bowers, L. Downey, M. Jones, B. Ludäscher, D. Pennington, A. Rajasekar, S. Romanello, M. Schildhauer, D. Vieglais, J. Zhang, Data integration and workflow solutions for ecology, Data Integration in the Life Sciences (2005) 321–324.

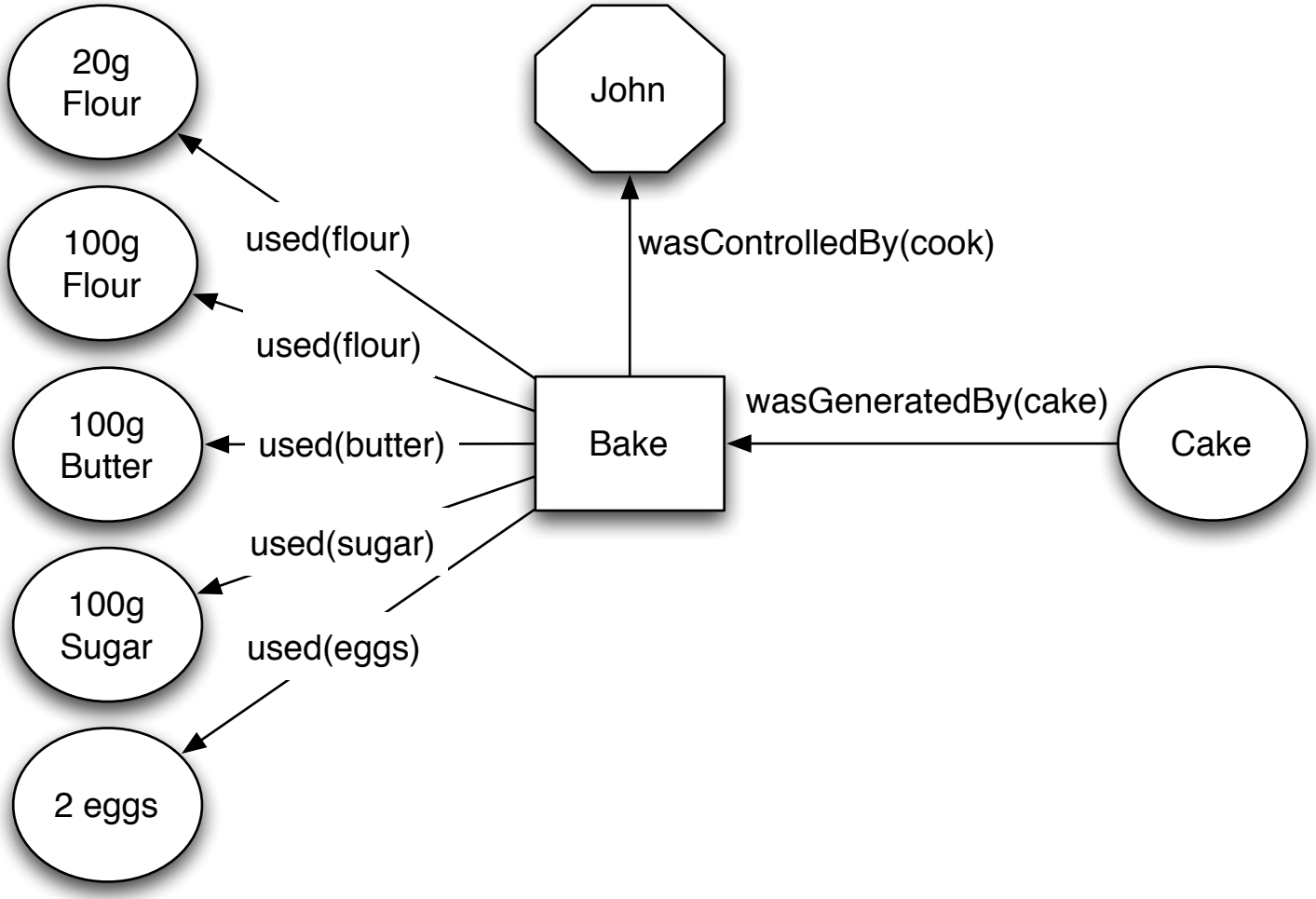
[52] K. Anyanwu, A. Maduko, A. Sheth, Sparq2l: Towards support for subgraph extraction queries in rdf databases, in: WWW '07: Proceedings of the 16th international conference on World Wide Web, ACM, New York, NY, USA, 2007, pp. 797–806. doi:http://doi.acm.org/10.1145/1242572.1242680.

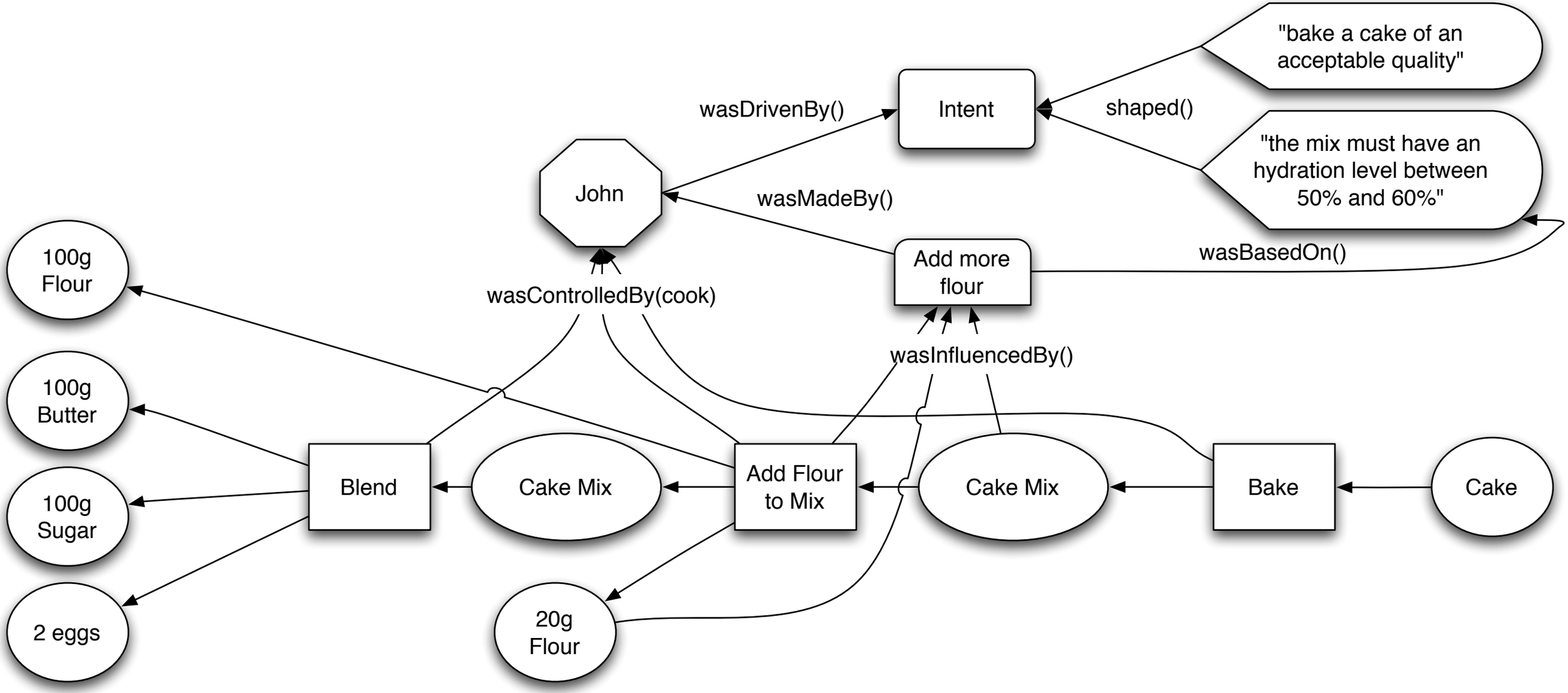
[53] T. Vitvar, A. Mocan, M. Kerrigan, M. Zaremba, M. Zaremba, M. Moran, E. Cimpian, T. Haselwanter, D. Fensel, Semantically-enabled service oriented architecture : concepts, technology and application, Service Oriented Computing and Applications 1 (2) (2007) 129–154. doi:10.1007/s11761-007-0009-9.

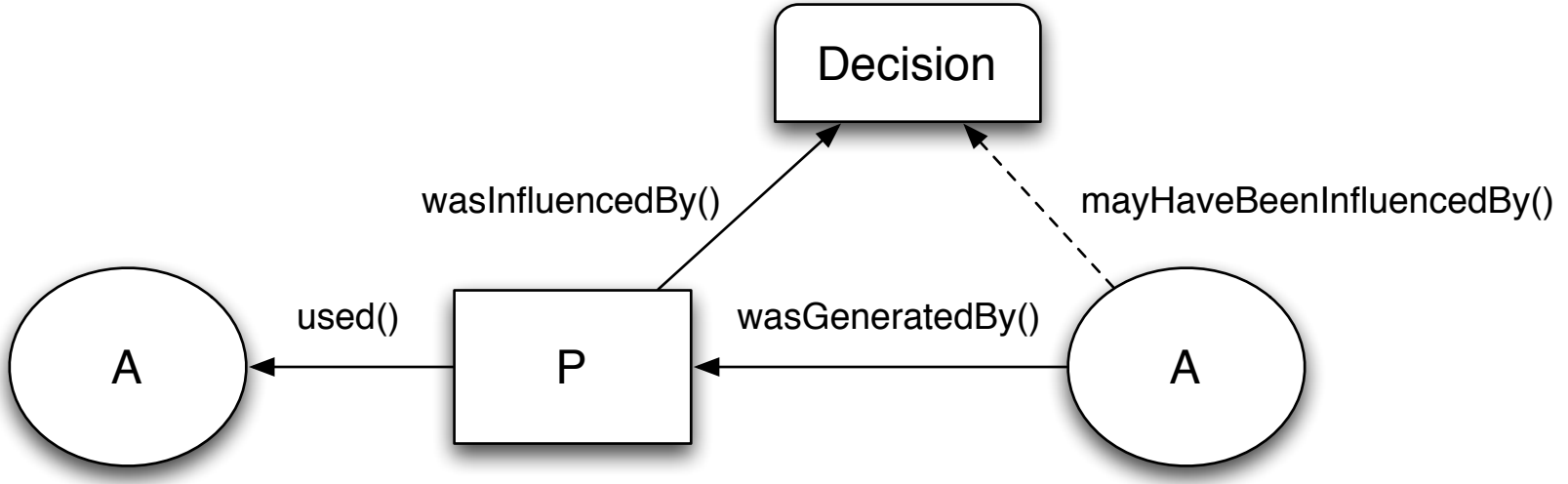
[54] F. Hielkema, P. Edwards, C. Mellish, J. Farrington, A flexible interface to community-driven metadata, in: Proceedings of the eSocial Science conference 2007, Ann Arbor, Michigan 2007, 2007.

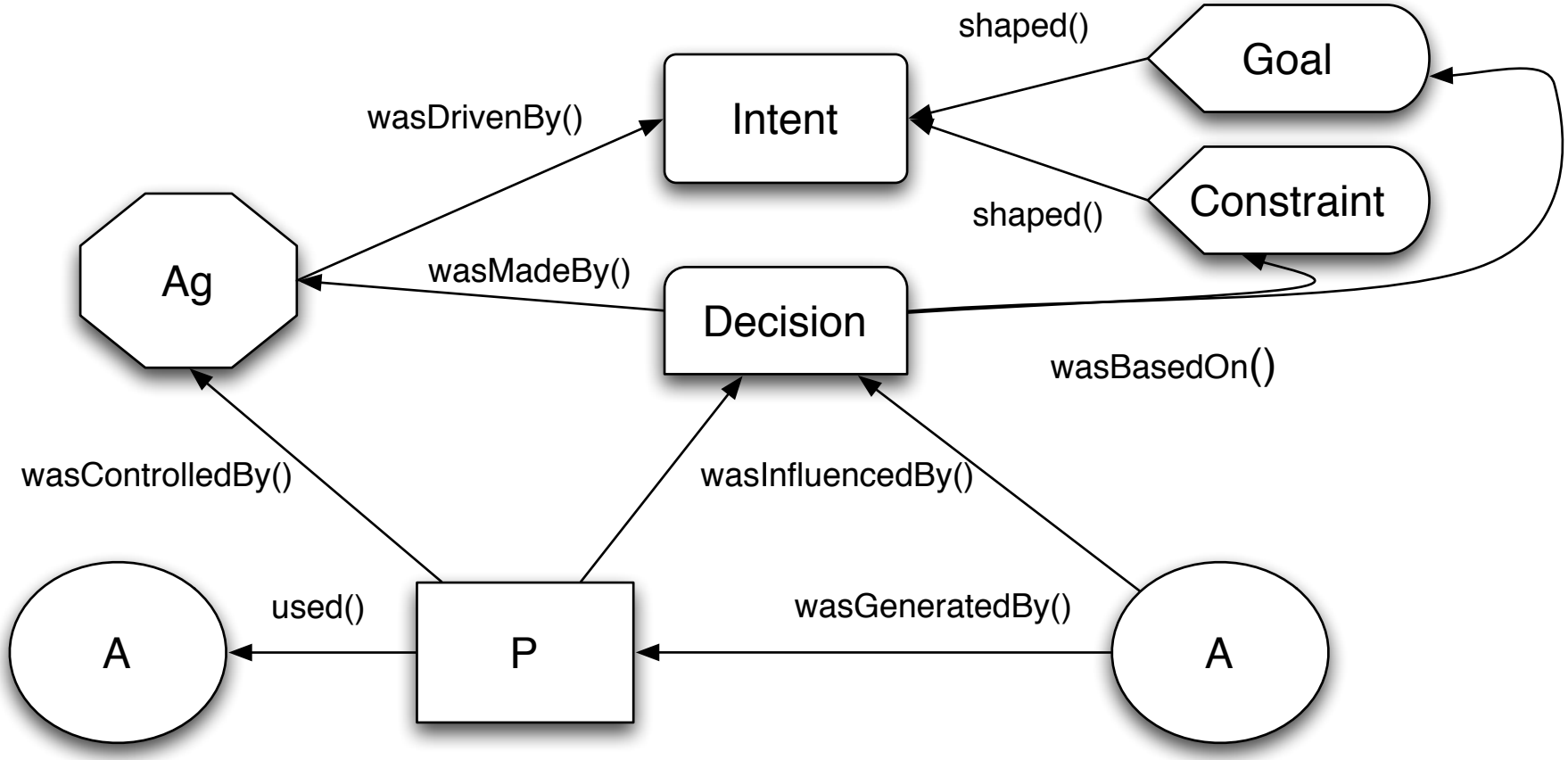
[55] J. Futrelle, J. Myers, Tracking provenance semantics in heterogeneous execution systems., Concurrency and Computation: Practice and Experience 20 (5) (2008) 555–564.

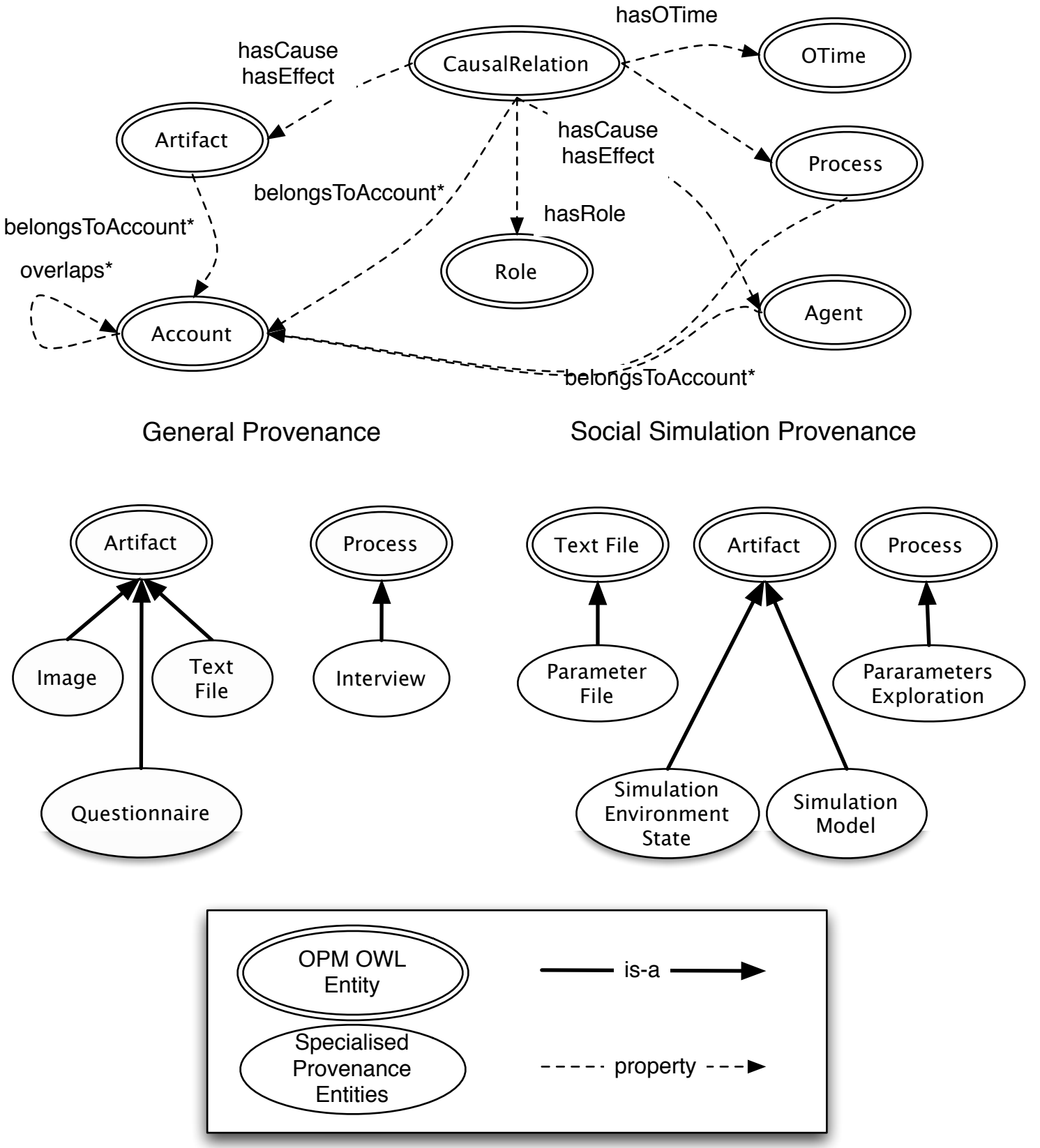

Workflow Experiment
WasControlledBy*
Workflow Engine
Used
WasGeneratedBy
opm:Agent
WasControlledBy*
role:Service
opm:Process
SI Agent
role:input
supportsAction*
Computational Resource
role:output
belongsToAccount*
WasDrivenBy*
Workflow State
Intent
wasSatisfiedOn*
belongsToAccount*
WasInfluencedBy
Shaped*
Shaped*
opm:Artifact
opm:Account
ruleml:imp
WasMadeBy
Constraint
Goal
WasInfluencedBy
WasBasedOn
WasBasedOn
hasPreCondition
hasPreCondition
Decision
Pre Condition = ruleml:head
Workflow Action
Action = ruleml:body
<COLLECTION>
<COLLECTION>
swrl:Atom
OPM OWL Entity
Intent
- property ►
is-a
role:service
CausalRelation


20g
Flour
John
100g
Flour
used(flour)
wasControlledBy(cook)
used(flour)
100g
Butter
used(butter)
Bake
wasGeneratedBy(cake)
Cake
used(sugar)
100g
Sugar
used(eggs)
2 eggs


"bake a cake of an acceptable quality"
Intent
shaped()
wasDrivenBy()
"the mix must have an hydration level between 50% and 60%"
John
wasMadeBy()
Add more flour
wasBasedOn()
100g Flour
wasControlledBy(cook)
wasInfluencedBy()
100g Butter
100g Sugar
2 eggs
Blend
Cake Mix
Add Flour to Mix
Cake Mix
Bake
Cake
20g Flour


Decision
wasInfluencedBy()
mayHaveBeenInfluencedBy()
A
used()
P
wasGeneratedBy()
A


shaped()
Goal
wasDrivenBy()
Intent
shaped()
Constraint
Ag
wasMadeBy()
Decision
wasBasedOn()
wasControlledBy()
wasInfluencedBy()
A
used()
P
wasGeneratedBy()
A


OPM Provenance
hasOTime
hasCause
hasEffect
CausalRelation
OTime
Artifact
hasCause
hasEffect
Process
belongsToAccount*
hasRole
belongsToAccount*
overlaps*
Role
Agent
Account
belongsToAccount*
General Provenance
Social Simulation Provenance
Artifact
Process
Image
Text File
Interview
Questionnaire
Text File
Artifact
Process
Parameter File
Pararameters Exploration
Simulation Environment State
Simulation Model
OPM OWL Entity
is-a
Specialised Provenance Entities
property