

Wander Antunes Gaspar Valente

**SciProv: uma Arquitetura para a Busca Semântica em
Metadados de Proveniência no Contexto de e-Science**

Dissertação apresentada ao Programa de Pós-graduação em Modelagem Computacional, da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do grau de Mestre em Modelagem Computacional.

Orientadora: Profa. D.Sc. Regina Maria Maciel Braga Villela

Juiz de Fora

2011

Valente, Wander Antunes Gaspar.

SciProv: uma arquitetura para a busca semântica em metadados de
proveniência no contexto de e-Science / Wander Antunes Gaspar Valente.
– 2011.

175 f. : il.

Dissertação (Mestrado em Modelagem Computacional)—Universidade
Federal de Juiz de Fora, Juiz de Fora, 2011.

1. Ciência da computação. 2. Processamento eletrônico de dados. I.
Título.

CDU 681.3

Wander Antunes Gaspar Valente

**SciProv: uma Arquitetura para a Busca Semântica em Metadados de
Proveniência no Contexto de e-Science**

Dissertação apresentada ao Programa de Pós-graduação em Modelagem Computacional, da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do grau de Mestre em Modelagem Computacional.

Aprovada em 18 de Janeiro de 2011.

BANCA EXAMINADORA

Profa. D.Sc. Regina Maria Maciel Braga Villela - Orientadora
Universidade Federal de Juiz de Fora

Prof. D.Sc. Leonardo Guerreiro Azevedo
Universidade Federal do Estado do Rio de Janeiro

Profa. D.Sc. Fernanda Cláudia Alves Campos
Universidade Federal de Juiz de Fora

*Dedico este trabalho
à Julia e ao Augusto.*

AGRADECIMENTOS

À Rita, pelo incentivo e apoio em todos os momentos.

Ao Wanderley Gaspar (*in memoriam*) e à Pequetita.

À professora Regina Braga, exemplo de competência profissional e de dedicação à educação de qualidade.

À professora Fernanda Campos pela valiosa contribuição.

À coordenação e ao corpo docente do Programa de Pós-Graduação em Modelagem Computacional.

À Universidade Federal de Juiz de Fora.

À todos aqueles que contribuíram para a concretização deste sonho.

*“Transportai um punhado de
terra todos os dias e fareis uma
montanha.” – Confúncio*

RESUMO

SciProv: uma Arquitetura para a Busca Semântica em Metadados de Proveniência no Contexto de e-Science

A *e-Science* se caracteriza pela manipulação de um vasto volume de dados e utilização de recursos computacionais em larga escala, muitas vezes localizados em ambientes distribuídos. Nesse cenário, representado por alta complexidade e heterogeneidade, torna-se relevante o tratamento da proveniência de dados, que tem por objetivo descrever os dados que foram gerados ao longo da execução de um experimento científico e apresentar os processos de transformação pelos quais foram submetidos. Assim, a proveniência auxilia a formar uma visão da qualidade, da validade e da atualidade dos dados produzidos em um ambiente de pesquisa científica.

O *SciProv* consiste em uma arquitetura cujo objetivo é interagir com sistemas de gerenciamento de *workflows* científicos para promover a captura e a gerência dos metadados de proveniência gerados. Para esse propósito, o *SciProv* adota uma abordagem baseada em um modelo abstrato para a representação da proveniência. Esse modelo, denominado *Open Provenance Model*, confere ao *SciProv* a capacidade de prover uma infraestrutura homogênea e interoperável para a manipulação dos metadados de proveniência. Como resultado, o *SciProv* permite disponibilizar um arcabouço para consulta às informações de proveniência geradas em um cenário complexo e diversificado de *e-Science*.

Mais importante, a arquitetura faz uso de tecnologia *web* semântica para processar as consultas aos metadados de proveniência. Nesse contexto, a partir do emprego de ontologias e máquinas de inferências, o *SciProv* provê recursos para efetuar deduções sobre os metadados de proveniência e obter resultados importantes ao extrair informações adicionais além daquelas que encontram-se registradas de forma explícita nas informações gerenciadas.

Palavras-chave: e-Science. Proveniência de Dados. Workflow Científico. Web Semântica. Open Provenance Model.

ABSTRACT

SciProv: An Architecture for Semantic Search on Provenance Metadata in e-Science Context

E-Science is characterized by manipulation of huge data set and large scale computing resources usage, often located in distributed environments. In this scenario, represented by high complexity and heterogeneity, it becomes important to treat data provenance, which aims to describe data that were generated during a scientific experiment execution and presents processes of transformation by which underwent. Thus, lineage helps to form a quality, validity and topicality vision of data produced in a scientific research environment.

SciProv consists of an architecture that aims to interact with scientific workflows management systems for capture and manipulation of generated provenance metadata. For this purpose, *SciProv* adopts an approach based on an abstract model for representing the lineage. This model, called Open Provenance Model, provides to *SciProv* the ability to set up a homogeneous and interoperable infrastructure for handling provenance metadata. As a result, *SciProv* is able to provide a framework for query data provenance generated in a complex and diverse e-Science scenario.

More important, the architecture makes use of semantic web technology to process metadata provenance queries. In this context, using ontologies and inference engines, *SciProv* provides resources to make inferences about lineage and to obtain important results in allowing the extraction of information beyond those that are registered explicitly from managed data.

Keywords: e-Science. Data Provenance. Scientific Workflow. Semantic Web. Open Provenance Model.

SUMÁRIO

1	INTRODUÇÃO	15
1.1	Motivação	15
1.2	Objetivo	18
1.3	Estrutura do Trabalho	21
2	PROVENIÊNCIA DE DADOS CIENTÍFICOS	23
2.1	Introdução	23
2.2	Workflow Científico	24
2.2.1	<i>Sistemas de Gerenciamento de Workflows Científicos</i>	24
2.3	Ciclo de Vida de um Experimento Científico	27
2.4	Caracterização da Proveniência de Dados	28
2.4.1	<i>Proveniência Prospectiva e Retrospectiva</i>	29
2.4.2	<i>Anotações</i>	29
2.5	Considerações de Projeto para uma Abordagem de Proveniência ...	30
2.5.1	<i>Modelo de Proveniência</i>	31
2.5.2	<i>Instrumentalização</i>	31
2.6	Mecanismos de Captura de Proveniência	32
2.6.1	<i>Mecanismo de Captura no Nível de Workflow</i>	32
2.6.2	<i>Mecanismo de Captura no Nível de Processo</i>	33
2.6.3	<i>Mecanismo de Captura no Nível de Sistema Operacional</i>	35
2.6.4	<i>Trabalhos Relacionados à Captura da Proveniência</i>	36
2.7	Modelo de Proveniência	39
2.7.1	<i>Grafo de Proveniência</i>	39
2.7.2	<i>Modelos de Proveniência com Abstrações em Camadas</i>	40
2.8	Modelo para Armazenamento e Consulta	42
2.8.1	<i>Modelo para Armazenamento</i>	42
2.8.2	<i>Modelo para Consulta</i>	43
2.8.3	<i>Modelos de Consulta Baseados em Tecnologia Web Semântica</i> ...	45
2.9	Novas Aplicações para os Dados de Proveniência	47

3	OPEN PROVENANCE MODEL	49
3.1	Requisitos do Open Provenance Model	49
3.2	Descrição do Modelo OPM	50
3.2.1	<i>Artefato, Processo e Agente</i>	51
3.2.2	<i>Dependências Causais</i>	52
3.2.2.1	<i>Papéis</i>	53
3.2.2.2	<i>Dependência Causal was controlled by</i>	53
3.2.2.3	<i>Dependência Causal was derived from</i>	53
3.2.2.4	<i>Dependência Causal was triggered by</i>	54
3.2.2.5	<i>Convenção para Relações de Causalidade</i>	54
3.2.2.6	<i>Dependências Causais Múltiplas</i>	54
3.2.2.7	<i>Dependências Causais Suportadas</i>	55
3.2.3	<i>Objetivo dos Papéis no Modelo OPM</i>	57
3.2.4	<i>Exemplos</i>	58
3.3	Descrições Sobrepostas e Hierárquicas	60
3.4	Restrições Temporais e Observação do Tempo	64
3.5	Compleitude e Inferências	67
3.5.1	<i>Regras de Compleitude</i>	67
3.6	Anotações	69
3.6.1	<i>Arcabouço de Anotações</i>	69
3.6.2	<i>Propriedades Fundamentais do OPM</i>	70
3.7	Perfis OPM	72
4	SCIENTIFIC WORKFLOW PROVENANCE SYSTEM	74
4.1	Arquitetura do <i>SciProv</i>	74
4.1.1	<i>Componentes da Arquitetura</i>	75
4.1.2	<i>Mecanismo de Instrumentalização</i>	77
4.1.3	<i>Componente Composto</i>	79
4.1.4	<i>Instrumentalização em Diversos Níveis de Detalhamento</i>	79
4.1.5	<i>Importância do Mecanismo de Coleta em Camadas</i>	81
4.1.6	<i>Persistência dos Dados de Proveniência</i>	82
4.1.7	<i>Geração do Grafo de Proveniência</i>	87
4.1.8	<i>Processo de Vinculação em Tecnologia Web Semântica</i>	89

4.1.9	<i>Vinculação em Formato RDF</i>	90
4.1.10	<i>Consulta com Recursos da Web Semântica</i>	92
4.1.11	<i>Especializações ao Modelo OPM</i>	94
4.2	<i>Implementação da Arquitetura do SciProv</i>	97
4.2.1	<i>Implementação do Mecanismo de Instrumentalização</i>	98
4.2.2	<i>Regra de Completude Inválida</i>	101
4.2.3	<i>Instrumentalização em Dois Níveis de Detalhamento</i>	103
4.2.4	<i>Implementação da Persistência Relacional</i>	107
4.2.5	<i>Implementação do Grafo de Proveniência</i>	108
4.2.6	<i>Serialização em Formatos XML e DOT</i>	110
4.2.7	<i>Serialização em Formato RDF</i>	111
4.2.8	<i>Implementação da Consulta ao Grafo de Proveniência</i>	114
4.2.9	<i>Consulta Semântica no SciProv</i>	115
4.2.10	<i>Consulta Semântica com Perfil Adaptado do modelo OPM</i>	117
5	ESTUDOS DE CASO	120
5.1	Banco de Dados de DNA do Japão	121
5.1.1	<i>Serviço GetEntry do DDBJ</i>	122
5.1.2	<i>Estudo de Caso com o SGWfC Kepler</i>	124
5.1.3	<i>Estudo de Caso com o SGWfC Taverna</i>	131
5.1.4	<i>Avaliação de Desempenho de Workflows Instrumentalizados</i>	135
5.1.5	<i>Estudo de Caso em Conjunto com os SGWfCs Taverna e Kepler</i>	138
5.1.6	<i>Estudo de caso com Arcabouço de Anotações OPM</i>	141
5.1.7	<i>Estudo de caso com Agregação de Ontologia do Domínio</i>	144
6	CONSIDERAÇÕES FINAIS	148
	REFERÊNCIAS	154
	APÊNDICES	161

LISTA DE ILUSTRAÇÕES

1.1	Exemplo de <i>workflow</i> científico modelado computacionalmente	19
1.2	Proveniência em um contexto de múltiplas aplicações com baixo acoplamento	20
2.1	Exemplo de <i>workflow</i> científico modelado no SGWfC Vistrails	25
2.2	Exemplo de <i>workflow</i> científico modelado no SGWfC Kepler	25
2.3	Exemplo de <i>workflow</i> científico modelado no SGWfC Taverna	26
2.4	Ciclo de vida de um experimento científico [30]	27
2.5	Proveniência prospectiva e retrospectiva [32]	30
2.6	Grafo de Proveniência [4]	40
2.7	Modelos de proveniência em camadas [9]	41
2.8	Modelos de proveniência em camadas [32]	42
2.9	Consulta à proveniência de dados a partir de uma interface gráfica [9]	45
2.10	Estrutura em camadas da <i>web</i> semântica [48]	46
3.1	Entidades e dependências no modelo OPM [54]	52
3.2	Exemplo de grafo de proveniência segundo o modelo OPM [54]	59
3.3	Grafo de proveniência “Bolo Assado pelo João” [54]	59
3.4	Grafos de proveniência em diferentes níveis de detalhamento [54]	61
3.5	Descrições sobrepostas e hierárquicas em um grafo de proveniência [54]	62
3.6	Refinamentos sucessivos em um grafo de proveniência [54]	63
3.7	Composição de um ciclo a partir de múltiplas descrições [54]	64
3.8	Tempo no modelo OPM [54]	66
3.9	Relacionamentos do tipo <i>ocorreu antes</i> no modelo OPM [54]	67
3.10	Compleitude: introdução e eliminação de artefatos [54]	68
3.11	Compleitude: introdução de processo [54]	69
3.12	Anotações em um grafo de proveniência [54]	70
4.1	Arquitetura do <i>SciProv</i>	75
4.2	Mecanismo de instrumentalização do <i>SciProv</i>	78
4.3	Abstração do Mecanismo de Instrumentalização	80
4.4	Exemplo de componente composto em dois níveis de abstração	81

4.5	<i>SciProv</i> – DER que representa o modelo conceitual	83
4.6	<i>SciProv</i> – Tabelas do modelo lógico	84
4.7	<i>SciProv</i> – Tabela do arcabouço de anotações	85
4.8	Arquitetura em camadas do modelo OPM	88
4.9	Exemplo de grafo simples expresso na linguagem DOT	90
4.10	Representação gráfica gerada a partir da linguagem DOT	90
4.11	Modelo em camadas do <i>SciProv</i> para a representação da proveniência	91
4.12	Componentes do conhecimento representado na arquitetura do <i>SciProv</i>	92
4.13	Hierarquia de classes da ontologia do OPM	93
4.14	Consultas semânticas mais ricas a partir de perfis especializados	95
4.15	Propriedade <i>derivedFromArtifact</i> da ontologia do OPM	95
4.16	Processamento semântico a partir da especialização do modelo OPM	96
4.17	Página inicial do <i>SciProv</i>	98
4.18	<i>Workflow</i> científico modelado no SGWfC Kepler	99
4.19	Instrumentalização de componente de um <i>workflow</i> no SGWfC Kepler	100
4.20	Grafo de Proveniência gerado no <i>SciProv</i>	102
4.21	Regra de Completude inválida no OPM	102
4.22	Componente composto modelado no SGWfC Kepler	103
4.23	Instrumentalização de componente composto – descrição <i>black</i>	104
4.24	Grafo de Proveniência OPM com descrições distintas	105
4.25	Instrumentalização em nível fino de detalhamento – descrição <i>red</i>	106
4.26	Tuplas da tabela <i>Causal_Dependency</i> do <i>SciProv</i>	107
4.27	Interface gráfica do <i>SciProv</i> para consultas SQL	108
4.28	Resultado da consulta SQL na interface com o usuário do <i>SciProv</i>	108
4.29	Interface com o usuário para seleção do <i>workflow</i> a ser representado na memória	109
4.30	Serialização em formato XML do grafo de proveniência OPM	111
4.31	Representação visual do grafo de proveniência OPM	112
4.32	<i>TBox</i> do modelo OPM em OWL-DL	113
4.33	Grafo de proveniência OPM serializado em RDF/XML	114
4.34	Interface com o usuário do <i>SciProv</i> para consulta em SPARQL	116
4.35	Resultado da consulta SPARQL ao grafo de proveniência em RDF	116
4.36	Aspecto da especialização da ontologia do modelo OPM	118

4.37	Relações de Dependência Inferidas no Grafo de Proveniência	119
4.38	Interface com o usuário do <i>SciProv</i> com resultados inferidos	119
5.1	Representação abstrata do <i>workflow</i> científico	123
5.2	Informações de cabeçalho de <i>Synechococcus elongatus</i>	124
5.3	Sequenciamento genético de <i>Synechococcus elongatus</i> (parte inicial)	125
5.4	<i>Workflow</i> para sequenciamento genético no Kepler	125
5.5	<i>Workflow</i> instrumentalizado para sequenciamento genético no Kepler	126
5.6	Metadados de proveniência gerados em sintaxe XML	127
5.7	Metadados de proveniência gerados em sintaxe RDF/XML	128
5.8	Grafo de proveniência do <i>workflow</i> científico modelado no Kepler	129
5.9	Características inversa e transitiva para a propriedade <i>triggeredProcess</i>	130
5.10	Pesquisa semântica em SPARQL no <i>workflow</i> científico modelado no Kepler	130
5.11	Resultado da busca semântica no <i>workflow</i> científico com inferências em destaque	131
5.12	<i>Workflow</i> científico modelado no SGWfC Taverna	132
5.13	Informações de cabeçalho de <i>Synechococcus elongatus</i> no Taverna	132
5.14	Sequenciamento genético de <i>Synechococcus elongatus</i> no Taverna	132
5.15	<i>Workflow</i> instrumentalizado para sequenciamento genético no Taverna	133
5.16	Grafo de proveniência do <i>workflow</i> científico modelado no Taverna	134
5.17	Resultado da busca semântica no <i>workflow</i> científico modelado no Taverna	135
5.18	Parte do <i>workflow</i> científico conjunto modelado no Taverna	139
5.19	Parte do <i>workflow</i> científico conjunto modelado no Kepler	140
5.20	Grafo de proveniência do <i>workflow</i> científico modelado de forma conjunta	142
5.21	Resultado da busca semântica no <i>workflow</i> científico modelado de forma conjunta	142
5.22	Grafo de proveniência com anotações OPM	144
5.23	SPARQL para busca semântica com base em anotações OPM	145
5.24	Resultado da busca semântica com base em anotações OPM	145
5.25	Ontologia simples referente à taxonomia parcial do organismo <i>Synechococcus</i> <i>elongatus</i>	146
5.26	Consulta SPARQL para busca semântica pelo Gênero e Família	147
5.27	Resultado da busca semântica por Gênero e Família	147

1 INTRODUÇÃO

1.1 Motivação

A partir das últimas décadas do século XX, com o progresso contínuo da Informática, o processo de concepção, execução e análise de experimentos científicos conduzidos por pesquisadores de diversas áreas tem buscado o apoio de ferramentas computacionais de forma crescente. Percebe-se também que o emprego de tais recursos está relacionado ao aumento na complexidade dos experimentos desenvolvidos que, em alguns casos, podem tornar-se inviáveis sem algum tipo de suporte computacional adequado [1]. Entretanto, cabe ponderar que os benefícios oferecidos por essas ferramentas computacionais implicam em novos e complexos desafios no cenário da pesquisa científica.

Nesse contexto, é possível apontar algumas questões importantes. Um aspecto fundamental está relacionado ao reuso do conhecimento gerado. Os experimentos científicos processados computacionalmente são passíveis de frequentes atualizações, seja por uma visão mais refinada dos pesquisadores ou por modificações em etapas e tarefas constituintes, que podem tornar-se de maior ou menor relevância ao refinar-se o modelo. Além disso, bancos de dados podem ser atualizados periodicamente, deixando defasados os resultados obtidos anteriormente. Por conseguinte, o reuso do conhecimento pode significar economia de recursos e de tempo dispendido pela comunidade científica.

Outro aspecto relevante refere-se aos resultados de um experimento processado computacionalmente, que poderá ter pequena utilidade se os cientistas não forem capazes de julgar a adequabilidade ao problema analisado e a identificar a origem dos resultados. Em um cenário de pesquisa científica, pode-se ponderar que parte do significado dos dados obtidos deve-se ao entendimento do processo gerador.

As questões levantadas nesses parágrafos iniciais conduzem ao tema da *proveniência de dados*, cuja caracterização pode ser descrita conforme se segue.

A *proveniência de dados* — também chamada de linhagem, genealogia ou *pedigree* — consiste na descrição das origens de um item de dado e do processo pelo qual foi produzido. A proveniência de dados auxilia a formar uma visão da qualidade, da validade e da atualidade a cerca da informação produzida no contexto de um experimento científico

modelado computacionalmente [2].

Em outros termos, a proveniência de dados fornece informações históricas referentes aos dados manipulados a partir das fontes originais [3]. Essa informação progressiva tem por objetivo descrever os dados que eventualmente foram gerados ao longo da execução do experimento e apresentar os processos de transformação aos quais esses dados foram submetidos.

Para os cientistas, a proveniência em experimentos científicos pode indicar como os resultados foram derivados, quais parâmetros tiveram influência no processo de derivação e quais bases de dados foram utilizadas como entrada para o processamento [4].

A partir da caracterização do conceito de proveniência, pode-se argumentar que as informações de linhagem têm potencial para agregar valor de forma significativa no processo de gerência e análise dos resultados de experimentos científicos obtidos através de simulações computacionais. Esta hipótese constitui o principal aspecto motivador para o desenvolvimento do presente trabalho.

Entretanto, para se obter os benefícios advindos a partir da proveniência, torna-se fundamental que as informações relacionadas à simulação computacional que representa o experimento científico sejam capturadas, modeladas e persistidas de forma adequada para posterior utilização. Nesse contexto, observa-se que o gerenciamento da proveniência de dados é uma questão em aberto e que tem merecido considerável tratamento por parte da comunidade científica [5] [6] [7] [8] [9].

Entre os problemas estudados pela comunidade de proveniência de dados é importante citar a falta de concordância quanto ao escopo e ao tratamento das informações a serem capturadas em um modelo de proveniência [10].

Nesse cenário, o presente trabalho baseia-se em um modelo de proveniência denominado *Open Provenance Model* (OPM) que tem por objetivo definir uma representação genérica e abrangente para a proveniência de dados, além do escopo de experimentos científicos [11].

É possível observar na literatura um esforço por parte de um grupo de pesquisadores envolvidos com o tema da proveniência de dados em aprimorar o OPM. A meta do grupo é construir e disponibilizar um modelo padrão *de fato* para o tratamento da proveniência que permita a interoperabilidade entre os diversos sistemas de *workflows* científicos já implementados ou que venham a ser projetados para a coleta e a gerência de informações

a cerca da linhagem dos dados.

Nesse contexto, argumenta-se nesse trabalho que a adoção pela comunidade científica do OPM como modelo padrão para o tratamento da proveniência é uma possibilidade a ser considerada, hipótese fundamentada a partir do contínuo desenvolvimento do modelo, que é apoiado por projetos relevantes na área de *workflows* científicos como Vistrails [12], Kepler [13] e Taverna [14].

A questão da proveniência pode ser relacionada com o ciclo de vida de um experimento científico e com a linhagem dos dados que emergem da partir de cada etapa desse processo [15]. Nesse contexto, o ciclo de vida de um *workflow* modelado computacionalmente compõem-se de três etapas: composição, execução e análise.

- A *composição* é a fase responsável pela concepção dos processos que constituem o experimento. Os dados de proveniência *prospectiva* (ver Seção 2.4.1) estão associados a essa etapa [9]. O mecanismo de coleta dos dados de proveniência deve ser capaz de capturar também o encadeamento do fluxo dos processos e atividades modeladas na composição do experimento bem como as dependências existentes entre os dados de entrada e de saída de todos os componentes envolvidos.
- Durante a fase de *execução* devem ser coletadas as informações relativas à proveniência *retrospectiva* [9], que pode ser interpretado como um *log* detalhado do histórico de execução da simulação computacional.
- Por último, na etapa de *análise*, é possível avaliar os resultados obtidos a partir da execução do experimento. Nesse ponto, o cientista pode promover alterações na especificação do modelo em estudo a partir de consultas aos dados de proveniência coletados, com o propósito de refinar ou corrigir o curso da pesquisa científica em andamento [16].

Deve-se ainda mencionar um outro aspecto importante a ser considerado na coleta e gerência da proveniência de dados – A questão do *nível de granularidade*. Esta pode ser entendida como o grau de detalhamento dos dados capturados. Esse aspecto da proveniência encontra-se diretamente associado à utilidade das informações de linhagem disponibilizadas para o pesquisador [11]. Observa-se na literatura científica diversas soluções para o problema da proveniência de dados que cobrem uma ampla faixa de granularidade [9].

O presente trabalho está inserido na linha de pesquisa em *e-Science* do Núcleo de Engenharia do Conhecimento (NEnC) da Universidade Federal de Juiz de Fora (UFJF),

com ênfase em estudos para a construção de uma infraestrutura de apoio a projetos neste domínio a partir de tecnologias como ontologias, serviços *web* semânticos e agentes. Nesse contexto, os estudos aqui desenvolvidos procuram avançar em temas abordados em trabalhos publicados anteriormente pelo NEnC, incluindo-se o arcabouço *CelOWS* [17] e as arquiteturas *SASAgent* [18] e *QDA ontology* [19].

Por fim, é válido mencionar que o tema desenvolvido no presente trabalho encontra respaldo a partir das diretrizes formuladas no documento *Grandes Desafios* [20] elaborado sob os auspícios da Sociedade Brasileira de Computação (SBC), particularmente em dois dos cinco tópicos propostos: (i) gestão da informação em grandes volumes de dados distribuídos; (ii) modelagem computacional de sistemas complexos artificiais, naturais e sócio-culturais e da interação homem-natureza.

1.2 Objetivo

Para que seja possível discorrer de forma mais clara sobre o objetivo do presente trabalho é importante apresentar alguns conceitos relevantes a cerca do tema abordado.

Um *experimento científico* engloba a modelagem, a execução e a análise de processos científicos de forma encadeada e controlada. Um experimento científico apoiado em tais premissas visa fundamentalmente a reprodutibilidade. Em outros termos, o objetivo dessa abordagem é permitir o rastreamento da cadeia de processos que deram origem a um determinado resultado e ainda facilitar o compartilhamento e a compreensão do experimento pela comunidade científica [1].

Nesse cenário, um *workflow* consiste em uma descrição de um processo reprodutível composto por um conjunto de tarefas inter-relacionadas [21]. Essa definição é válida no contexto de experimentos científicos, onde um *workflow* pode ser entendido como um fluxo de dados processado ao longo de uma cadeia de atividades [22].

Um *workflow* representa a orquestração de uma sequência de processos que manipulam dados de modo a construir uma simulação e pode ser considerado o principal recurso de um experimento científico [15]. A Figura 1.1 apresenta um exemplo de *workflow* científico modelado a partir de uma ferramenta computacional específica para esse fim. O objetivo desse *workflow* científico consiste em buscar e tratar as informações referentes ao sequenciamento genético de um organismo e disponíveis em um repositório distribuído de

dados.

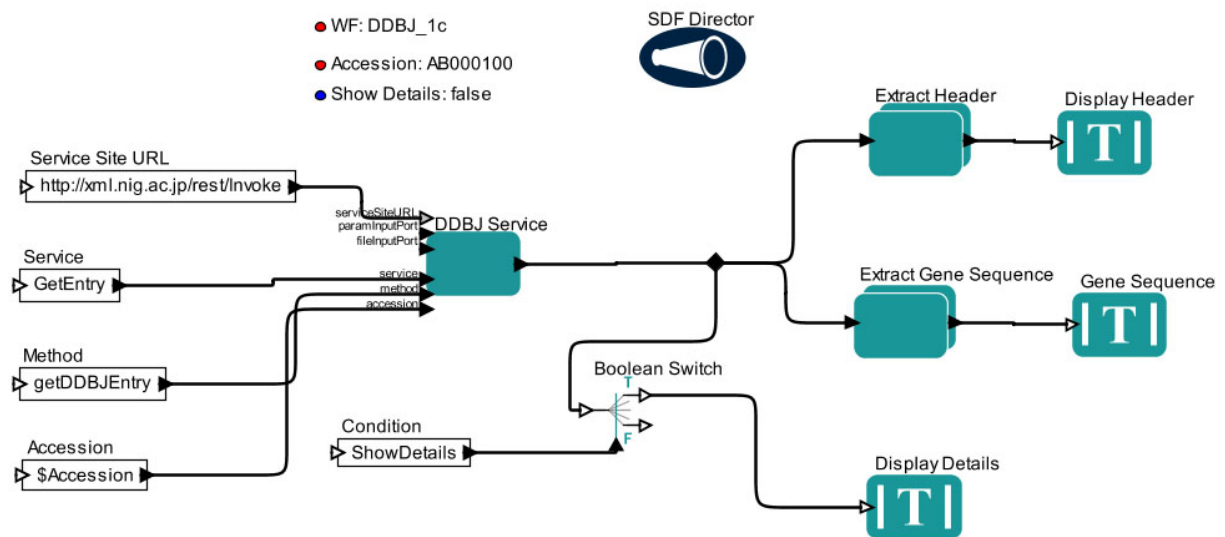


Figura 1.1: Exemplo de *workflow* científico modelado computacionalmente

Cabe observar que em um *workflow* científico a ênfase reside, em geral, em um grande volume de dados a ser processado. Essa característica implica na necessidade de um robusto e sofisticado aparato computacional para oferecer o suporte adequado à execução. Nesse contexto, diversos *Sistemas de Gerência de Workflows Científicos* (SGWfC) têm sido desenvolvidos com o propósito de prover um ferramental adequado aos processos de orquestração e de execução dos experimentos de natureza científica.

O objetivo de um SGWfC pode ser entendido como um esforço dirigido à comunidade científica no sentido de construir e aprimorar um ambiente computacional integrado para simplificar o esforço necessário para modelar e gerir um *workflow* científico [23].

Um *ambiente de pesquisa colaborativo disperso geograficamente* pode ser entendido como uma tendência de organização entre grupos científicos com interesse em áreas de estudo correlatas. Esse modelo de relacionamento pode incluir mecanismos de interação social, técnicas de colaboração, modelos de comunicação formal e informal e acordos sobre normas, princípios, valores e regras [24].

Em um contexto de pesquisa científica o que se observa é o desenvolvimento desses ambientes impulsionado pela necessidade de se compartilhar dados e também conhecimento sobre os dados. Observa-se que as informações compartilhadas podem ser mais úteis em um contexto que possibilite ao colaboradores a compreensão necessária para um uso efetivo dos dados [25].

Para viabilizar um efetivo ambiente de pesquisa em ambiente colaborativo disperso

geograficamente é fundamental dispor de recursos de interconexão de dados a partir de uma *grade computacional*. Esse modelo de integração pode ser entendido como uma infraestrutura de *hardware* e de *software* capaz de prover acesso seguro, consistente, pervasivo, de alto desempenho e de baixo custo a recursos computacionais sofisticados [26].

Nesse contexto, caracterizado por aplicações multi-institucionais com baixo acoplamento e componentes heterogêneos, como é possível determinar a proveniência dos dados manipulados? A Figura 1.2 ilustra a situação vislumbrada. O que se pretende nesse trabalho é tratar a questão da interoperabilidade para a proveniência de dados em um cenário de pesquisa colaborativa conforme descrito.

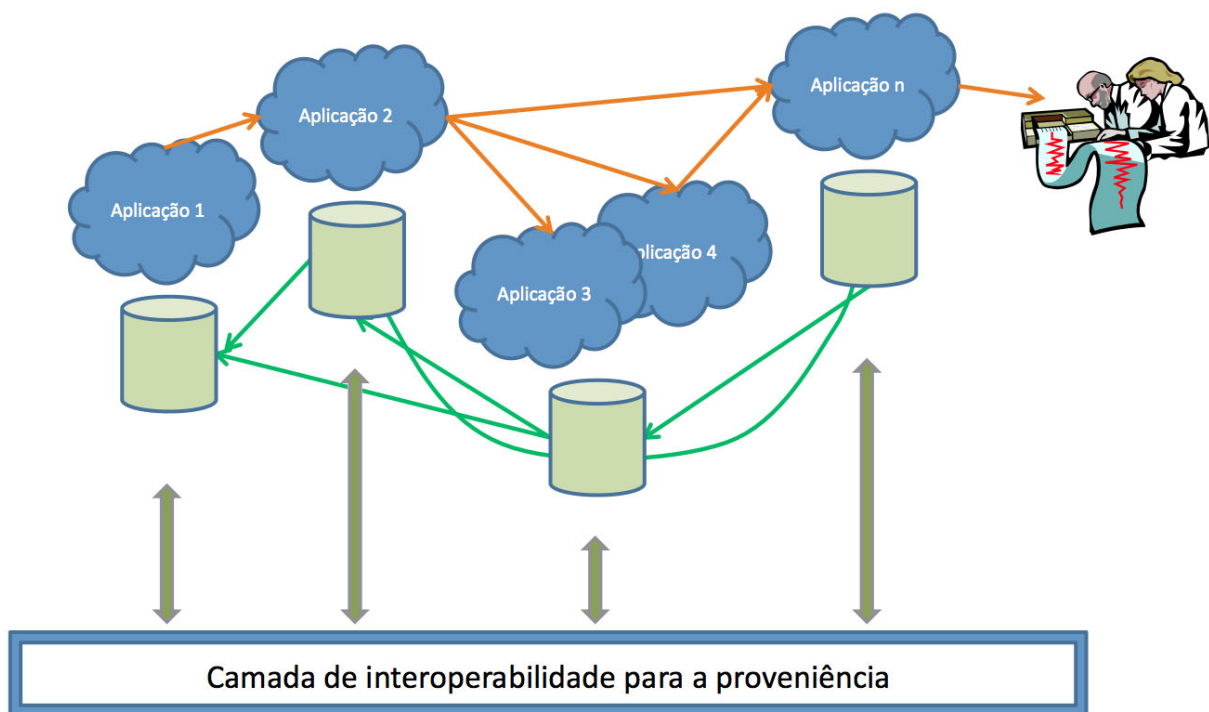


Figura 1.2: Proveniência em um contexto de múltiplas aplicações com baixo acoplamento

Pode-se argumentar que o tratamento da proveniência no contexto de uma aplicação específica encontra-se em um estágio à frente. SGWfC como Kepler, Taverna e Vistrails oferecem recursos para a coleta, registro e consulta aos metadados de proveniência gerados a partir de um *workflow* científico [27]. Entretanto, cabe considerar o tratamento da linhagem em um cenário mais amplo, conforme apresentado na Figura 1.2. O objetivo é manipular a proveniência a partir dos dados gerados por múltiplas aplicações.

Nesse sentido, o presente trabalho baseia-se em uma visão aberta de proveniência (*Open Provenance Vision*) proposta pelo modelo OPM, que consiste em prover o suporte adequado para que a proveniência de dados capturada a partir de sistemas heterogêneos

possa ser expressa, reunida de forma coerente e consultada de uma maneira integrada [27]. Assim, a visão do OPM provê um conjunto de guias de estilo e melhores práticas para o apoio à interoperabilidade no contexto da proveniência de dados.

Com base nessas definições torna-se possível apresentar o objetivo do presente trabalho, que consiste em especificar uma arquitetura para a coleta e gerência da proveniência de dados e processos no contexto de experimentos científicos processados através de simulações computacionais em ambientes de pesquisa colaborativa dispersos geograficamente e interconectados através de uma grade computacional. A arquitetura proposta deve prover uma camada de interoperabilidade capaz de interagir com os SGWfC tendo como finalidade capturar as informações de proveniência gerados a partir de *workflows* científicos processados computacionalmente.

A concepção do modelo a ser apresentado baseia-se em alguns requisitos considerados fundamentais ao delimitar-se o escopo do trabalho delineado, a saber:

1. Arquitetura independente dos mecanismos de controle de fluxo e formatos de dados implementados em quaisquer SGWfC existentes.
2. Aplicabilidade em uma ampla faixa de experimentos científicos, incluindo aqueles executados em ambientes heterogêneos — a partir de diversas plataformas de *hardware* e sistemas operacionais — e dispersos em uma grade computacional.
3. Uso de metodologias e de tecnologias no contexto atual da Ciência da Computação e, em particular, ao conhecimento científico diretamente relacionado com a área de proveniência de dados.
4. Avaliação do impacto da coleta, persistência e consulta aos metadados de proveniência sobre o desempenho dos experimentos científicos.

1.3 Estrutura do Trabalho

Esta dissertação está organizada em seis capítulos. O Capítulo 2 apresenta e discute a proveniência de dados em um contexto que abrange a concepção e a execução de experimentos científicos através de simulações computacionais. Em adição, são revistos os fundamentos necessários para o desenvolvimento do tema proposto. O Capítulo 3 discute em maiores detalhes o *Open Provenance Model* [11], modelo de proveniência empregado como base no desenvolvimento desse trabalho. O Capítulo 4 descreve a arquitetura para

a coleta e gerência da proveniência de dados e processos no domínio de experimentos científicos, denominada *Scientific Workflow Provenance System (SciProv)*. O Capítulo 5 apresenta um conjunto de estudos de caso em que o *SciProv* é empregado para coletar e gerenciar os metadados de proveniência de um experimento científico executado em um ambiente de pesquisa colaborativa e interconectado por uma grade computacional. Finalmente, o Capítulo 6 formula algumas conclusões pertinentes ao projeto e sugere alguns trabalhos futuros.

2 PROVENIÊNCIA DE DADOS CIENTÍFICOS

2.1 Introdução

Tradicionalmente a teoria e a experimentação científica constituem as bases do processo de investigação de fenômenos naturais ou artificiais, onde o objetivo consiste em adquirir novos conhecimentos ou corrigir e integrar conhecimentos previamente estabelecidos. Entretanto, as técnicas e ferramentas oriundas da Ciência da Computação têm proporcionado grandes transformações na prática contemporânea da pesquisa científica. Assim, nesse novo cenário, é válido reconhecer a computação como um importante ferramental para o avanço da Ciência, juntamente com a teoria e a experimentação [20].

Nesse contexto, o que se pode observar é uma Ciência que tem utilizado cada vez mais procedimentos computacionais com o intuito de lidar com o aumento constante no volume de dados e processos necessários à realização dos experimentos de pesquisa. Pode-se argumentar que apesar do conhecimento científico continuar a ser gerado por métodos tradicionais, *in-vivo* e *in-vitro*, novas modalidades de experimentos científicos vêm ganhando importância: *in-virtuo* e *in-silico* [15].

- Experimentos *in-virtuo* são conduzidos por cientistas e pesquisadores através de simulações computacionais utilizando-se de modelos numéricos. Entende-se por modelo numérico uma representação computacional de elementos ou fenômenos do mundo real. O comportamento do modelo é analisado através de simulação.
- Experimentos *in-silico* dizem respeito a ambientes de estudo unicamente compostos por modelos numéricos e computacionais e sem nenhuma interação humana.

A importância de experimentos científicos *in-virtuo* e *in-silico* está relacionada tanto à redução de custos e de riscos quanto à reprodutibilidade. Além disso, provêm um mecanismo de suporte às atividades experimentais tradicionais, *in-vivo* e *in-vitro* [28].

Contudo, pode-se ponderar que o cenário das simulações computacionais *in-virtuo* e *in-silico* apresenta características análogas aos primórdios da Computação, no sentido de que

os centros de pesquisa podem depender do conhecimento individual de pesquisadores para o encadeamento dos programas e *scripts* necessários para a execução dos experimentos formulados.

Esse processo tende a estar sujeito a falhas e a ser improdutivo, especialmente em casos de experimentos científicos de alta complexidade, que envolvam muitos programas, grande quantidade de dados e grupos de pesquisadores trabalhando em localidades geograficamente dispersas [1].

2.2 Workflow Científico

No contexto apresentado, surge o conceito de *workflow* científico como um paradigma para a representação e gestão de experimentos científicos complexos, cuja implementação computacional passa a ser utilizada com o objetivo de facilitar a abstração e permitir uma composição estruturada de programas e *scripts* como uma sequência de atividades que visa um determinado resultado [23].

2.2.1 *Sistemas de Gerenciamento de Workflows Científicos*

A ampliação da complexidade na abordagem dos problemas científicos aliado aos avanços na tecnologia como um todo e na Ciência da Computação em particular têm resultado no desenvolvimento de *Sistemas de Gerenciamento de Workflows Científicos*, que se constituem em um conjunto de ferramentas computacionais desenvolvidas para tornar a automação do processo científico mais eficiente e mais produtivo [23].

As Figuras 2.1, 2.2 e 2.3 ilustram a orquestração de *workflows* a partir dos SGWfCs Vistrails, Taverna e Kepler, que representam três soluções disponíveis interessantes no contexto da modelagem computacional de experimentos científicos [16].

Cada etapa de um *workflow* especifica um processo ou um artefato de *software* a ser executado (por exemplo, um serviço *Web*). O *workflow* provê a conexão entre as etapas do experimento modelado de acordo com o fluxo de dados e as respectivas dependências. A representação computacional de um *workflow* contém diversas informações importantes que podem ser utilizadas em um processo de análise e verificação de resultados, incluindo os processos que foram executados e os recursos de armazenamento utilizados em ambientes distribuídos [29].

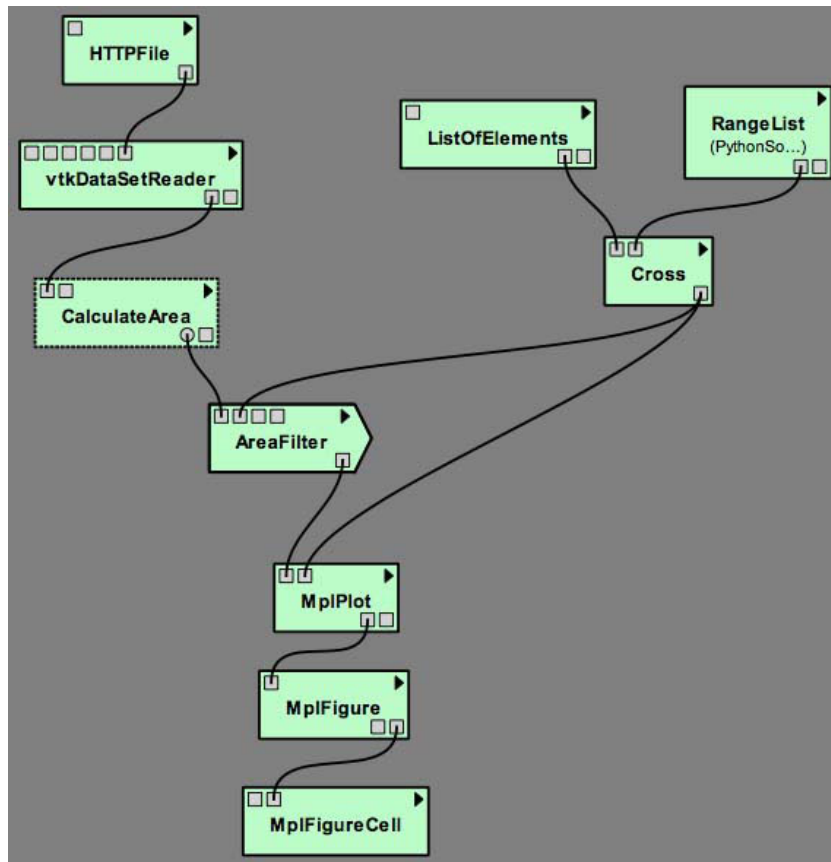


Figura 2.1: Exemplo de *workflow* científico modelado no SGWfC Vistrails

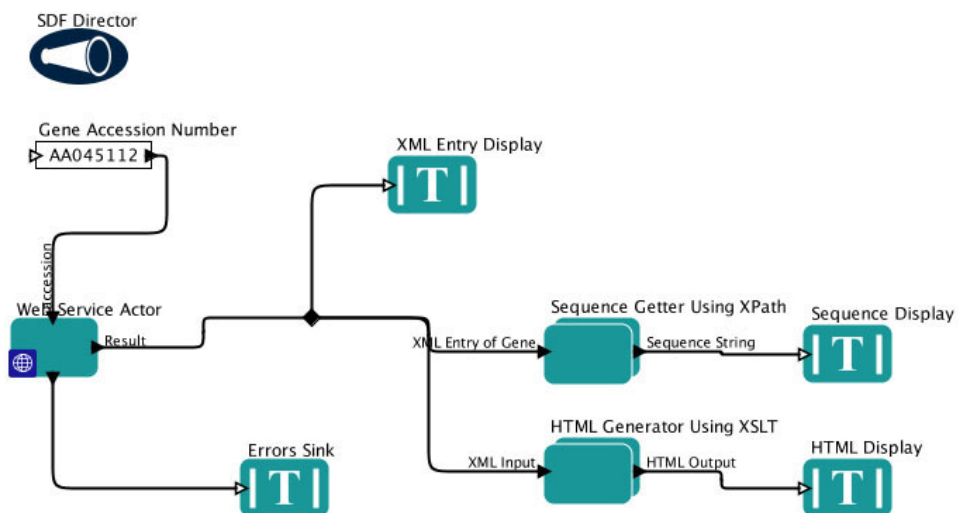


Figura 2.2: Exemplo de *workflow* científico modelado no SGWfC Kepler

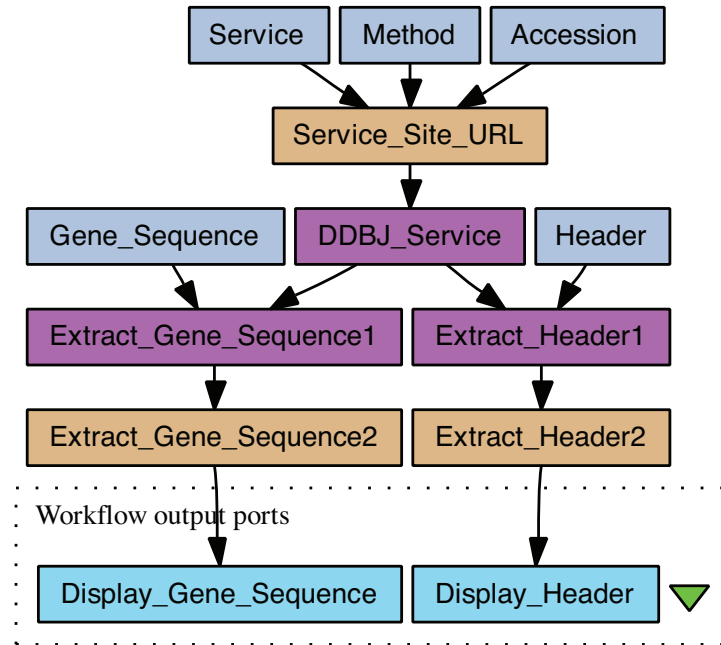


Figura 2.3: Exemplo de *workflow* científico modelado no SGWfC Taverna

Um SGWfC permite explorar as representações de processos computacionais complexos em diversos níveis de abstração, com o objetivo de gerenciar o ciclo de vida e automatizar a execução. Além disso, a automação de *workflows* pode fornecer as informações necessárias para a reprodutibilidade científica, a derivação de resultados e o compartilhamento de resultados em um ambiente de pesquisa colaborativo [5].

Um volume de pesquisa considerável encontra-se em andamento para tratar os problemas relativos à concepção e reutilização de *workflows* científicos, coleta e monitoramento de dados de proveniência, otimização de desempenho e confiabilidade [15].

É possível perceber que, para a plena realização das promessas lançadas pela tecnologia de *workflows*, é necessário atender a diversos requisitos e desafios surgidos a partir das próprias aplicações científicas. Nesse cenário, torna-se importante examinar questões como o suporte a experimentos orientados a eventos altamente dinâmicos e recursos para a interação dos usuários de forma a prover suporte ao desenvolvimento colaborativo e compartilhamento dos resultados obtidos [29].

Todo esse panorama apresentado tem por objetivo situar o presente trabalho como uma contribuição que se insere nos estudos relacionados a experimentos científicos processados através de simulações computacionais e, mais especificamente, no tratamento de questões referentes à proveniência de dados.

2.3 Ciclo de Vida de um Experimento Científico

Pode-se considerar que o emprego de *workflows* não constitui por si só uma solução completa de apoio aos experimentos *in-silico*. Em adição, o entendimento da questão deve ser pautado pela inserção do paradigma de *workflows* em um contexto mais abrangente no que tange à gestão de dados científicos.

Nesse cenário, insere-se o suporte ao registro dos dados produzidos pelo fluxo de trabalho realizado, incluindo como e porque cada informação obtida foi gerada [30]. A Figura 2.4 apresenta o ciclo de vida de um experimento científico *in-silico*. Esse modelo tem por objetivo possibilitar um melhor entendimento dos relacionamentos existentes no contexto da aplicação de *workflows* em ambientes de pesquisa científica.

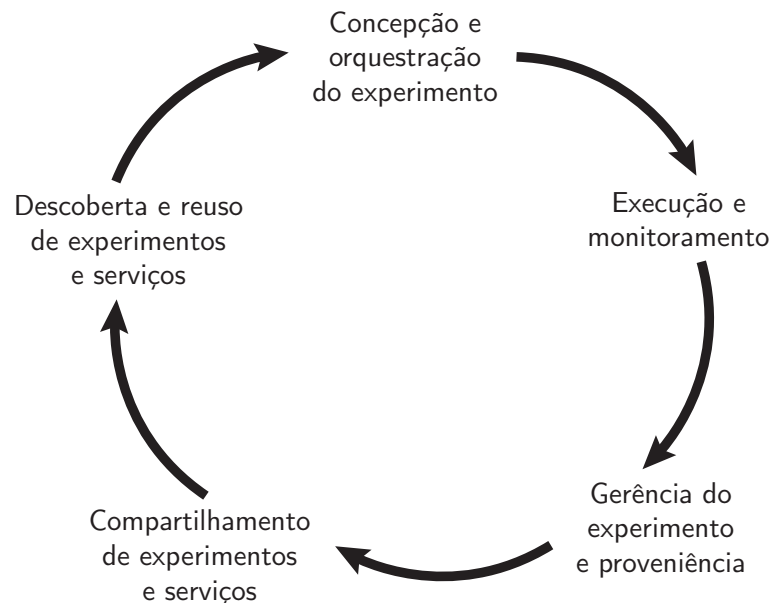


Figura 2.4: Ciclo de vida de um experimento científico [30]

Com base na Figura 2.4, é possível identificar os principais processos envolvidos no ciclo de vida de um experimento científico:

- *Concepção e Orquestração* – Refere-se à conceituação do escopo de cada uma das atividades componentes do *workflow* e à seleção de programas, *scripts* e de componentes adequados bem como da configuração do fluxo das atividades incluídas.
- *Execução e Monitoramento* – Com foco na execução do *workflow* propriamente dito, incluindo a distribuição dos componentes (dados e programas) e o monitoramento

da execução. Também está incluída nessa fase a gerência de configuração e a rastreabilidade da execução do *workflow*.

- *Gerência do Experimento e Proveniência* – O foco consiste na avaliação dos resultados obtidos a partir da execução do experimento, incluindo atividades como visualização de dados e consulta à proveniência.
- *Compartilhamento de Experimentos e Serviços* – Refere-se à disponibilização dos artefatos de *software*, *workflows* e resultados obtidos para grupos de pesquisa em ambiente colaborativo ou mesmo para a comunidade científica.
- *Descoberta e Reuso de Experimentos e Serviços* – Refere-se à busca para reutilização de artefatos de *software*, *workflows* e resultados disponibilizados por grupos de pesquisa em ambiente colaborativo ou mesmo pela comunidade científica.

2.4 Caracterização da Proveniência de Dados

Segundo o dicionário Oxford¹, o termo *proveniência* pode ser entendido como (a) a origem ou a história conhecida de alguma coisa; (b) a história ou a genealogia de uma obra de arte, um manuscrito ou uma antiguidade, etc.

Inicialmente, o problema da proveniência de dados foi caracterizado por Buneman [2], para o qual a proveniência de dados — também chamada de linhagem, genealogia ou *pedigree* — consiste na descrição das origens de um item de dado e do processo pelo qual foi produzido. Acrescenta-se que a proveniência auxilia a formar uma visão da qualidade, da validade e da atualidade a respeito das informações tratadas.

No contexto de sistemas computacionais, a proveniência pode ser entendida como um conjunto de processos que originou um determinado dado [4]. Por processos deve-se entender todas as derivações, conjuntos de dados, parâmetros, artefatos de *software*, componentes de *hardware* e modelos computacionais envolvidos na derivação do dado.

Ainda nesse contexto, pode-se ressaltar a existência de aplicações diversas relacionadas à proveniência de dados [3]. Os dados ancestrais podem, por exemplo, serem usados para estimar a qualidade e a confiabilidade dos resultados gerados bem como para permitir a realização de auditorias no sentido de verificar a real utilização dos dados de origem ou para detecção de fontes de erros e inconsistências.

¹Oxford Dictionary disponível em <http://www.askoxford.com/?view=uk>

Encontram-se na literatura revisões (*surveys*) técnicas que abordam o tema da proveniência de dados em um contexto de ambientes computacionais. Publicados ambos em 2005, Bose e Frew [31] fornecem uma visão abrangente dos primeiros trabalhos desenvolvidos sobre o assunto e Simmhan *et al.* [3] apresentam uma taxonomia sobre o tema e compara alguns sistemas de gerenciamento de *workflows* científicos segundo esse modelo de classificação.

Em outro trabalho mais atual, Cruz *et al.* [16] apresentam uma discussão interessante sobre um conjunto de características importantes encontradas em onze dos mais relevantes SGWfCs disponíveis e formula uma taxonomia com base nessas características.

Por fim, Freire [9] traça um paralelo entre diferentes abordagens para o problema da proveniência e aponta algumas questões em aberto, no sentido de facilitar a tomada de decisões ao se selecionar ou desenvolver uma solução no âmbito da pesquisa científica.

2.4.1 Proveniência Prospectiva e Retrospectiva

Davidson e Freire [32] consideram a existência de duas formas distintas de proveniência: *prospectiva* e *retrospectiva*. A proveniência prospectiva captura a especificação — o modelo — de uma tarefa computacional. Em outros termos, a proveniência prospectiva corresponde às etapas a serem seguidas para gerar um produto (dados). A proveniência retrospectiva relaciona-se à captura dos passos que foram executados bem como as especificações acerca do ambiente de execução.

A Figura 2.5 ilustra os dois tipos de proveniência propostos. A definição do *workflow* representa a proveniência prospectiva. A proveniência retrospectiva é ilustrada através dos recortes de dados (incluindo-se gráficos e imagens) gerados e coletados durante a execução do *workflow* e apresentados à esquerda e na parte inferior.

2.4.2 Anotações

A Figura 2.5 apresenta também uma classe adicional de proveniência que pode ser inserida até mesmo diretamente pelo usuário. Tal classe de proveniência, denominada *anotações*, tem por objetivo inserir informação adicional relevante para o entendimento do processo de orquestração e execução do *workflow* modelado.

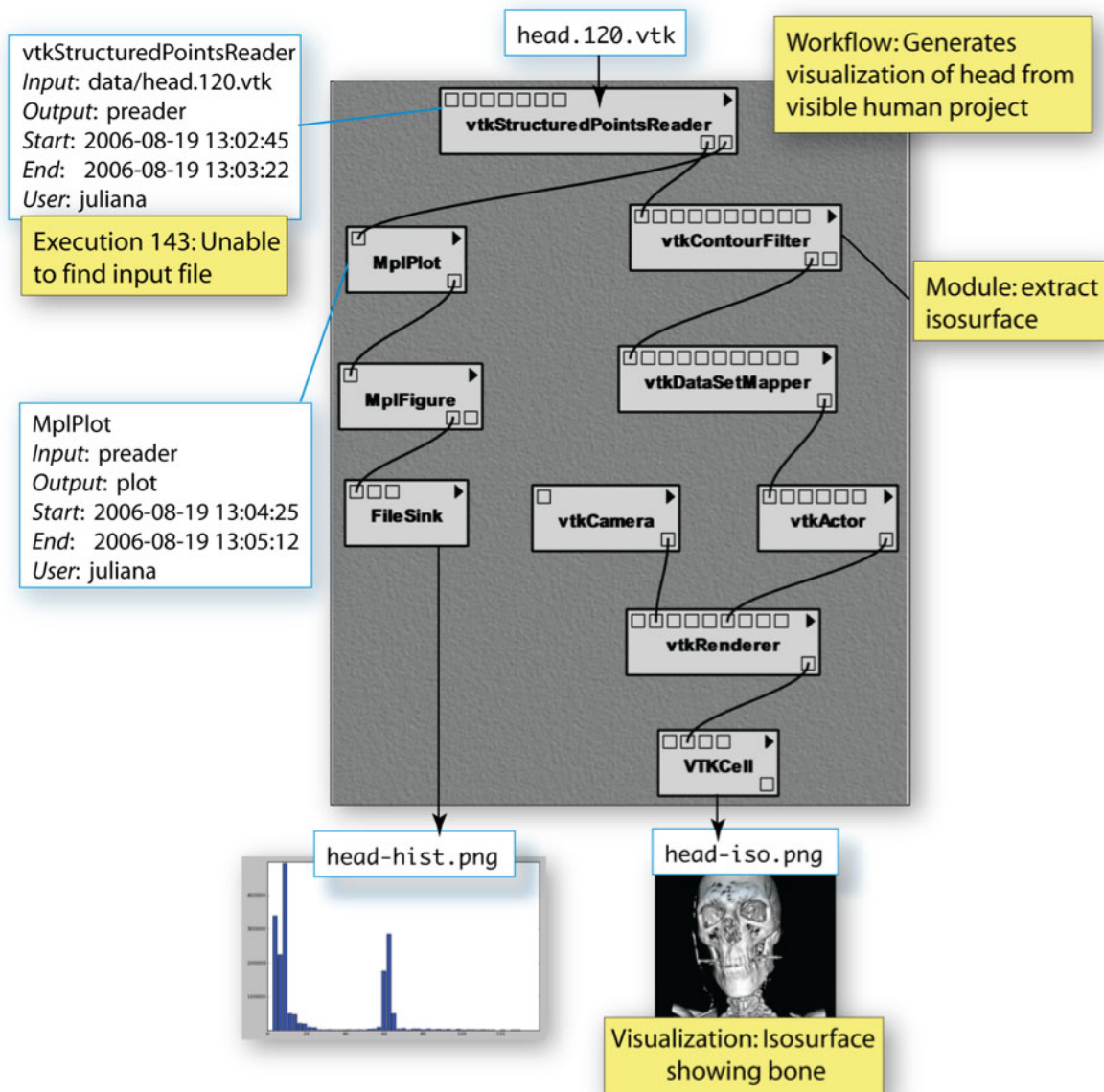


Figura 2.5: Proveniência prospectiva e retrospectiva [32]

2.5 Considerações de Projeto para uma Abordagem de Proveniência

As considerações de projeto para uma abordagem de proveniência referem-se às soluções propostas para problemas como coleta, gerenciamento e consulta à linhagem dos dados.

Nesse contexto, encontram-se na literatura científica trabalhos que defendem uma abordagem para a questão de forma independente de quaisquer sistemas gerenciadores de *workflows* existentes [7] [33].

Essa estratégia oferece uma alternativa à tendência dos SGWfC em disponibilizar os

recursos para o tratamento da proveniência diretamente integrados em tais ferramentas computacionais. Porém, é importante observar que essa integração resulta em um modelo de proveniência fortemente atrelado à arquitetura do SGWfC subjacente.

2.5.1 Modelo de Proveniência

Nesse cenário, o presente trabalho coloca em destaque o modelo de proveniência proposto por Moreau e colaboradores [11], cuja contribuição refere-se a especificação de um modelo padrão denominado *Open Provenance Model* (OPM). Tal proposta se diferencia dos demais modelos ao definir uma representação abstrata, genérica e abrangente para o tratamento da linhagem de dados.

O modelo OPM representa os dados de proveniência através de um *grafo de dependência causal*, que permite, por exemplo, registrar o histórico do fluxo de execução de um *workflow*. Deve-se considerar, entretanto, que o modelo OPM conforme especificado, não provê o suporte à representação de informações sobre a especificação do *workflow* e, por isso, pode ser classificado como um modelo de proveniência retrospectiva. Entretanto, o modelo também permite a representação da proveniência definida na forma de anotações.

2.5.2 Instrumentalização

Em uma abordagem de coleta e gerenciamento da proveniência de forma desvinculada de quaisquer sistemas de gerenciamento de *workflows* científicos existentes, outras considerações importantes requerem um tratamento adequado no sentido de se equacionar de forma mais conveniente os problemas enfrentados.

Torna-se preponderante a prescrição de um mecanismo de *instrumentalização*, cujo objetivo é permitir a coleta de dados de execução de experimentos científicos modelados a partir de diferentes SGWfCs. Estudos encontrados na literatura científica, como [7] e [6], têm como objetivo abordar o problema.

O foco dessas pesquisas consiste em especificar um modelo de coleta e armazenamento de dados de proveniência capaz de atender a duas premissas importantes nesse contexto: (a) permitir algum grau de automatização do processo de instrumentalização; (b) proporcionar um baixo impacto na performance de execução do *workflow* científico cujos dados de proveniência retrospectiva são coletados ao longo do processamento.

Essas premissas possuem relevância no âmbito do tratamento da proveniência em experimentos científicos. Por um lado, pode-se considerar que a maioria dos pesquisadores não possui a necessária proficiência em Computação ou a disponibilidade de tempo requerida para a implementação manual de mecanismos de instrumentalização necessários à coleta dos dados de proveniência gerados ao longo do processo de execução dos experimentos *in-silico*. Por outro lado, deve-se controlar a degradação imposta na performance de execução dos *workflows* a partir da incorporação dos mecanismos necessários à coleta e à gerência da proveniência.

2.6 Mecanismos de Captura de Proveniência

O mecanismo de captura é responsável pela coleta das informações de proveniência do *workflow*, que podem ser tanto referentes à proveniência prospectiva quanto à retrospectiva. Um mecanismo de captura deve ter acesso a detalhes de procedimentos computacionais, tais como artefatos de *software*, dados gerados em tempo de execução e anotações.

Os mecanismos de captura de proveniência podem atuar no nível do *workflow* como um todo, no nível de processo (atividade, artefato) ou no nível do sistema operacional subjacente. Mecanismos de captura baseados em *workflows* são aqueles integrados ou anexados ao SGWfC. Mecanismos de captura baseados em processo requerem que cada atividade definida em um *workflow* possa disponibilizar os dados de proveniência necessários. Por último, mecanismos de captura baseados no Sistema Operacional apóiam-se em funcionalidades existentes no sistema para gerar dados de proveniência.

2.6.1 Mecanismo de Captura no Nível de Workflow

A captura em nível de *workflow* constitui uma tendência importante e tem sido adotada por importantes SGWfCs tais como Kepler, Taverna e Vistrails. Ela constitui também a abordagem mais comum na literatura científica sobre o tema [9] [10].

Conforme já mencionado, uma característica importante dos mecanismos baseados em *workflow* refere-se ao forte acoplamento com o SGWfC subjacente, o que permite um processo de captura de dados de proveniência de forma direta. Nesse sentido, existem gerenciadores de *workflows* científicos capazes de processar a coleta automatizada das informações de proveniência.

Acrescenta-se que sistemas mais recentes, como Redux [34] e Vistrails [12] foram projetados para incluir o suporte nativo à proveniência no próprio modelo de arquitetura enquanto que os primeiros gerenciadores, tais como Taverna [14] e Kepler [13], receberam extensões posteriores para dar suporte a esses recursos [9].

Uma vez que os gerenciadores de *workflows* científicos têm acesso às definições do *workflow* e controlam a execução, podem, por conseguinte, capturar tanto dados de proveniência prospectiva quanto retrospectiva. A vantagem dessa estratégia de captura de dados de proveniência está ligada à facilidade de aplicação, uma vez que o mecanismo está acoplado diretamente ao SGWfC subjacente.

Deve-se considerar, porém, que dados gerados internamente por atividades específicas e que não são identificados na execução do *workflow* podem não ser capturados por esse tipo de mecanismo [10]. Um exemplo particularmente importante referente a essa restrição refere-se a *workflows* modelados a partir do encadeamento de serviços *web* executados remotamente. Nesse caso, tais artefatos de *software* comportam-se como “caixas-pretas” para o SGWfC, que, em princípio, não tem acesso ao processamento interno de tais componentes.

2.6.2 Mecanismo de Captura no Nível de Processo

Em um mecanismo de captura de proveniência que atue no nível de processo ou atividade, cada componente do *workflow* deve dispor de recursos que permitam a coleta das informações de linhagem.

Uma alternativa interessante consiste em instrumentalizar os componentes de um *workflow* modelado a partir de um SGWfC de forma que a proveniência possa ser coletada e posteriormente gerenciada de forma independente do gerenciador de *workflows*. Esse mecanismo é particularmente adequado nos casos em que o *workflow* que modela o experimento científico baseia-se fortemente no emprego de serviços *web*.

A captura dos dados de linhagem no nível de processo relaciona-se diretamente à proveniência retrospectiva [9]. Embora este aspecto possa ser considerado restritivo, por outro lado é possível identificar características desejáveis nessa estratégia para a coleta da proveniência.

Nesse contexto, considerando-se que o mecanismo de captura relaciona-se direta e especificamente com cada atividade presente no *workflow*, abre-se assim a possibilidade de

controle e seleção sobre os dados a serem coletados. Deve-se ressaltar que esta característica é particularmente importante porque relaciona-se diretamente à granularidade dos dados coletados. Entre as implicações do controle sobre os dados de proveniência coletados é importante mencionar três aspectos relevantes:

- Constitui um mecanismo para mensurar o impacto da coleta de dados na performance de execução de um experimento instrumentalizado. Em outros termos, o controle do nível de granularidade dos dados capturados pode ser empregado para avaliar e quantificar o desempenho do processo de coleta de proveniência durante a execução do *workflow* científico.
- Permite a coleta de proveniência em diversos níveis de detalhamento até mesmo de forma concomitante. Nesse aspecto, cada processo constituinte do *workflow* pode ser instrumentalizado diversas vezes, onde cada mecanismo de adaptação seja modelado de forma a capturar a linhagem em níveis de granularidade distintos.
- Em um ambiente de pesquisa colaborativo, torna-se viável o processo de adaptação de processos diretamente pelos proprietários do artefato de *software*. Essa possibilidade é particularmente interessante no caso de *workflows* modelados a partir de serviços *web* disponibilizados em um ambiente colaborativo. Considerando-se que em princípio um *web service* constitui uma “caixa-preta” para o usuário, essa alternativa viabiliza a adaptação de um artefato de *software* no sentido de prover recursos para a captura de dados relevantes no contexto do experimento científico em estudo.

Entretanto a instrumentalização dos componentes de um *workflow* requer a adaptação dos processos utilizados na modelagem do experimento científico, de forma a suportar as funcionalidades do mecanismo de coleta de dados [10]. Para *workflows* complexos ou quando a exigência do nível de granularidade for muito fina, a instrumentalização de forma manual pode impor um considerável trabalho de adaptação.

Nesse cenário, torna-se necessário considerar o tempo necessário para executar a instrumentalização do *workflow* bem como a disponibilidade de pessoal técnico com competência para desempenhar a tarefa a partir de gerenciadores de *workflows* científicos como Kepler ou Taverna.

Ainda no contexto da captura das informações de proveniência no nível de processo, o *Provenance for Recording Services – PreServ* [35] consiste em uma proposta para a captura

dos dados de proveniência no nível de processo em ambientes de execução distribuída e a integração das informações coletadas. Muito embora o foco do *PreServ* relaciona-se diretamente com o presente trabalho, deve-se observar entretanto que não utiliza o OPM como modelo abstrato para a proveniência de dados.

Por fim, vale mencionar o trabalho de Marinho *et al.* [10], que propõem uma abordagem interessante para a captura de dados de proveniência no nível de processo. Os autores investigam um mecanismo cujo objetivo consiste em processar a coleta de proveniência de forma automática e, por conseguinte, capaz de minimizar o trabalho de adaptação manual dos componentes de um *workflow* científico. Novamente, deve-se observar que o modelo de proveniência adotado em [10] não está relacionado ao OPM.

2.6.3 Mecanismo de Captura no Nível de Sistema Operacional

Mecanismos que atuam no nível do sistema operacional utilizam as funcionalidades disponíveis para capturar informações de proveniência. Em um cenário possível, a *Application Programming Interface* (API) do sistema de arquivos pode ser empregada para capturar os dados produzidos pelas atividades do *workflow* e descobrir as suas dependências.

Outra possibilidade refere-se à utilização do registro de chamadas do sistema operacional (SO) com o objetivo de identificar usuários e relacioná-los com as atividades executadas. Observa-se que os mecanismos de captura baseados no SO subjacente estão limitados ao tratamento dos dados de proveniência retrospectiva [36].

Nesse contexto, é possível registrar informações detalhadas sobre todas as chamadas de processos e arquivos acessados durante a execução de uma atividade. Entretanto, o grande número de chamadas geradas em nível do SO mesmo em processos simples pode tornar impraticável o registro e a manipulação de todos os dados de proveniência [37].

A captura de dados de proveniência no nível de SO constitui uma opção mais interessante em cenários de execução do *workflow* em ambientes distribuídos em razão da independência de quaisquer gerenciadores de workflows científicos. Entretanto, a fina granularidade dos dados coletados pode implicar na necessidade de adoção de algum esquema de pré-processamento, capaz de selecionar as informações pertinentes aos objetivos propostos no experimento científico em estudo, como, por exemplo, a ordem de execução das atividades e os respectivos dados gerados.

Para finalizar as considerações apresentadas sobre os mecanismos de coleta de prove-

niência, é importante exibir uma síntese dos requisitos desejáveis para a coleta de dados de proveniência em *workflows* científicos, conforme proposto por Simmham *et al.* [7]:

1. Existência de uma interface aberta e projetada com foco na interoperabilidade.
2. Arquitetura independente dos mecanismos de controle de fluxo e formatos de dados implementados nos SGWfC disponíveis.
3. Impacto mínimo tanto na necessidade de modificações impostas aos pesquisadores para a captura de proveniência quanto na correspondente sobrecarga gerada no processamento.

A presente proposta de trabalho procura manter-se aderente a tais premissas. Tanto o modelo de arquitetura quanto as diretrizes para a implementação de um protótipo, detalhados no Capítulo 4, encontram-se norteadas a partir desses referenciais básicos.

2.6.4 Trabalhos Relacionados à Captura da Proveniência

São encontrados na literatura científica trabalhos referentes à captura de dados de proveniência que adotam uma abordagem pelo menos em parte alinhada com os requisitos formulados por Simmham *et al.* [7], como Karma [7], Redux [38] e ProvManager [10].

O modelo proposto por Simmham *et al.* [7] tem como foco a captura da proveniência em *workflows* orquestrados a partir de recursos distribuídos em uma grade computacional baseados na invocação de serviços *web*.

O mecanismo de coleta proposto atua no nível de processo (atividade) e o conjunto de informações coletado é armazenado em um repositório central, que pode ser acessado através do *Karma Provenance Service*, um serviço *web* construído para esse fim. Nesse modelo, o pesquisador fica responsável por anexar um mecanismo de identificação para cada atividade do *workflow* desenvolvido. Esse mecanismo constitui a base para o processo de captura e publicação dos dados de proveniência do modelo proposto. A sobrecarga no desempenho de um *workflow* adaptado para a coleta de proveniência através do arcabouço que compõe o projeto foi estabelecida na ordem de 1% [7], de acordo com os parâmetros e estudos de caso apresentados.

Vale considerar que a arquitetura proposta no presente trabalho, denominada *Sci-Prov*, emprega um mecanismo de adaptação de *workflows* que segue um princípio análogo àquele proposto por Simmham *et al.* [7]. Além disso, o enfoque de ambos os trabalhos,

baseado em *workflows* modelados a partir de serviços *web* distribuídos em uma grade computacional, estabelece uma afinidade a cerca do escopo estudado para o problema da proveniência. Entretanto, as pesquisas desenvolvidas no presente trabalho trazem como diferencial a adoção do modelo de proveniência OPM e o emprego de tecnologia *web* semântica.

O trabalho de Groth *et al.* [33] insere-se em um amplo projeto denominado *The Provenance Project*², cujo objetivo é prover o suporte à proveniência de dados para problemas complexos executados em ambientes de grades computacionais. O estudo define um modelo de arquitetura de proveniência independente de implementações existentes e com foco no reuso. Essa arquitetura propõe um mecanismo de captura de proveniência em nível de atividade.

Alguns sistemas seguem o modelo de arquitetura de proveniência proposto por Groth *et al.*, [33] entre eles TENT [39] e EHCR [40]. Ressalta-se novamente que o diferencial do *SciProv* reside na adoção do modelo de proveniência OPM e no emprego de tecnologias oriundas da *web* semântica.

Barga e Digiampietri [38] argumentam que a captura dos dados de proveniência deve ser produzida de forma automatizada a partir de um SGWfC e gerida através de sistema subjacente de armazenamento. Segundo os autores, cada experimento científico possui necessidades distintas quanto à coleta de informações de proveniência. Assim, um modelo único e rígido para essa finalidade não seria capaz de satisfazer a requisitos específicos. O que esses pesquisadores propõem trata-se de um modelo flexível e em camadas de abstração para a coleta e o armazenamento da proveniência da execução de *workflows* científicos:

- *Abstract Workflow* – informações sobre a estrutura do *workflow*;
- *Model Workflow* – informações sobre a instanciação das atividades;
- *Executable Workflow* – informações sobre dados de entrada e parâmetros das atividades;
- *Runtime Workflow* – informações sobre a execução do *workflow*: quando cada atividade foi executada, onde, quais os dados gerados, exceções que ocorreram, etc.

Além dos dados de proveniência produzidos automaticamente e armazenados pelo Redux [34], o cientista pode inserir anotações sobre os dados, sobre as ferramentas e

²Disponível em <http://www.gridprovenance.org>

sobre os *workflows* em qualquer uma das camadas de proveniência.

O presente trabalho não estabelece um mecanismo de coleta automatizada de proveniência conforme proposto em [38]. Entretanto, o conceito de modelo de abstração em camadas é adotado na arquitetura do *SciProv*, uma vez que o padrão OPM contempla o estabelecimento de diversos níveis de instrumentalização para a coleta de dados.

A abordagem apresentada por Marinho *et al.* [10] visa dar suporte a um sistema de proveniência e monitoramento de execução de *workflows* científicos em ambientes distribuídos. O modelo proposto, denominado *ProvManager*, baseia-se em um mecanismo de captura de dados em nível de atividade, no qual o cientista deve proceder previamente a configuração do *workflow* para permitir o acoplamento com o sistema de proveniência.

Entretanto, alegam os autores — embora sem apresentarem resultados conclusivos —, essa configuração pode ser feita automaticamente pelo próprio *ProvManager*, que provê os recursos necessários para a adaptação de cada atividade do *workflow* no sentido de permitir o acoplamento com o mecanismo de captura de proveniência. O processo de configuração consiste em três etapas: (a) o pesquisador fornece a especificação do *workflow* para o configurador de proveniência; (b) o configurador instrumentaliza o *workflow* com o mecanismo de captura de proveniência; e (c) uma nova especificação do *workflow* adaptado é fornecida ao cientista.

O configurador de proveniência do *ProvManager* é extensível e permite a conexão com diversos SGWfC existentes, como Vistrails [12], Kepler [13] e Taverna [14], através de adaptadores específicos para cada um desses sistemas.

Conforme já mencionado, o presente trabalho não estabelece um mecanismo de coleta automatizada de proveniência. Entretanto, o *SciProv* permite a conexão com diversos SGWfCs, incluindo Vistrails, Kepler e Taverna. Nesse sentido, a arquitetura proposta emprega um único artefato de *software* — um serviço *web* — para estabelecer um mecanismo de instrumentalização adaptável aos SGWfCs citados.

Groth *et al.* [4] propõem um modelo de proveniência denominado *Pipeline-Centric Provenance Model*. A abordagem proposta baseia-se no conceito de Sistema de Dados Virtuais (*Virtual Data System*), onde parte das informações de proveniência não são persistidas.

Assim, em alguns casos, as consultas à proveniência requerem a re-execução do *workflow* para a obtenção dos dados históricos necessários. Essa estratégia tem como objetivo

melhorar a eficiência no armazenamento dos dados de proveniência e tem como foco simulações computacionais onde o volume de informações processadas é muito elevado como, por exemplo, no domínio da Astronomia.

2.7 Modelo de Proveniência

Um modelo de proveniência tem por objetivo definir as informações que devem ser suportadas em uma abordagem de proveniência. O presente trabalho avaliou diversos modelos encontrados na literatura científica que propõem soluções para a representação da proveniência [41], [42], [38], [43], [11] e [4]. A definição do modelo de proveniência adotado pela arquitetura baseou-se na aderência a alguns critérios considerados fundamentais, com destaque para a questão do tratamento da interoperabilidade dos metadados de linhagem.

Em um contexto de pesquisa colaborativa em ambientes heterogêneos e distribuídos, é preponderante que o modelo de proveniência ofereça o suporte ao tratamento consistente e homogêneo dos dados de linhagem. O modelo escolhido, denominado *Open Provenance Model* (OPM) é focado na interoperabilidade e, em acréscimo, fornece um conjunto de guias de estilo e melhores práticas para o desenvolvimento de aplicações interoperáveis no contexto da proveniência de dados. Vale também mencionar como itens que pesaram na escolha do OPM o desenvolvimento contínuo do padrão e a adoção por parte dos principais SGWfCs como Vistrails, Kepler e Taverna [27].

Segundo Groth *et al.* [4], para determinar a proveniência dos dados gerados por um sistema computacional, é preciso verificar a relação entre as etapas envolvidas no processo, identificar como os passos intermediários foram executados e quais dados foram utilizados em cada etapa durante a execução.

2.7.1 Grafo de Proveniência

Esta informação pode ser modelada como um grafo que conecta os dados gerados ao respectivo componente que o gerou. Este, por sua vez, deve estar conectado aos dados de entrada, que também está ligado a outro elemento e assim por diante (Figura 2.6).

Assim, o objetivo é obter um grafo de proveniência que reproduza fielmente a execução da simulação computacional em questão. É importante observar que os usuários podem formular consultas sobre a proveniência de qualquer parte do grafo — dados intermediários

— e não apenas sobre os resultados finais.

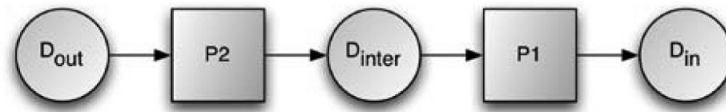


Figura 2.6: Grafo de Proveniência [4]

2.7.2 Modelos de Proveniência com Abstrações em Camadas

Segundo Simmhan *et al.* [3], a maioria dos modelos propostos baseia-se em protocolos proprietários para gerenciar a proveniência e, portanto, não fundamentam a coleta, representação, armazenamento e consulta em padrões abertos e consensuais. Em comum, todos oferecem suporte à proveniência retrospectiva e muitos à proveniência prospectiva e também a anotações [9].

Vale mencionar que a diversidade de modelos de proveniência existentes dificulta a interoperabilidade entre os metadados obtidos a partir de diferentes SGWfCs. Uma possível solução consiste na implementação de tradutores capazes de viabilizar a intercomunicação entre modelos. Porém, como não existe um padrão “de fato” estabelecido, cada modelo de proveniência deve construir tradutores específicos para prover o intercâmbio de dados e informações [10].

Especificamente, o modelo de proveniência implementado pelo Vistrails é organizado em três camadas [12]. A camada *Workflow Evolution* tem por objetivo capturar os relacionamentos entre os diversos fluxos de trabalho criados em uma tarefa exploratória; a camada *Workflow* trata individualmente as especificações dos fluxos de trabalhos e a camada *Execution* armazena os dados em tempo de execução referentes aos módulos do *workflow*. Essa estrutura em camadas conduz a uma representação normalizada que tem por objetivo estabelecer um relacionamento entre os diversos tipos de dados de proveniência capturados — por exemplo, conectar uma definição de *workflow* ao seu conjunto de execuções — e a evitar redundâncias no armazenamento de informações.

O modelo de proveniência proposto em Redux [38] também é configurado em camadas. A primeira camada corresponde a uma descrição abstrata do fluxo de trabalho, em que cada módulo corresponde a uma classe de atividades. Esta descrição abstrata está vinculada a serviços específicos e conjuntos de dados definidos na segunda camada. A

terceira camada captura informações sobre dados de entrada e parâmetros necessários ao processamento do *workflow* e a quarta camada captura detalhes em tempo de execução, tais como horário de início e término.

Também em Pegasus [44] é adotado um modelo em camadas para o tratamento dos metadados de proveniência. Considerando-se que este SGWfC possui como foco o processamento de *workflows* científicos modelados em uma grade computacional, torna-se interessante distinguir informações relativas ao agendamento de procedimentos em *grid* (camada *Workflow Instance*) e à execução (camada *Executable Workflow*).

A Figura 2.7 apresenta alguns modelos de proveniência com abstrações em camadas.

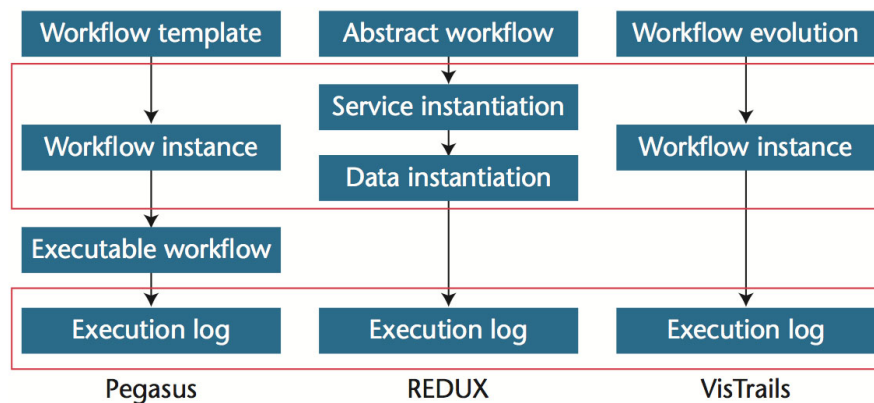


Figura 2.7: Modelos de proveniência em camadas [9]

Pode-se observar que os modelos de proveniência propostos a partir dos sistemas de gerência de *workflows* científicos tendem a se adaptar ao domínio e às necessidades dos usuários. Assim, apesar dos modelos esforçarem-se para atender a generalizações, os casos de uso específicos acabam por influenciar no projeto.

Nesse cenário, os modelos de proveniência do Taverna [14] e Redux [38] foram desenvolvidos inicialmente para prover o suporte a experimentos na área de Bioinformática e incluem uma infraestrutura que emprega ontologias de domínio específicas. Vistrails foi projetado para prover suporte a processos exploratórios nos quais os *workflows* são refinados iterativamente e, por conseguinte, utiliza um modelo de proveniência concebido com o objetivo de capturar metadados relativos à evolução do experimento [9].

O Capítulo 3 detalha o modelo abstrato de proveniência denominado *Open Provenance Model* (OPM). Este modelo tem por objetivo definir uma representação genérica e abrangente para a proveniência de dados [11]. No contexto do presente trabalho, o OPM é empregado como base conceitual para a coleta e a gerência da linhagem de dados.

2.8 Modelo para Armazenamento e Consulta

As pesquisas em proveniência têm ampliado a abrangência de questões em estudo, incluindo trabalhos com foco na infraestrutura para armazenamento, acesso e consulta aos dados armazenados. As implicações de avanços nesses campos de pesquisa vão desde uma melhor exploração e compreensão dos resultados até novas estratégias de reuso das informações [32].

O que se vislumbra é a possibilidade do usuário identificar *workflows* capazes de serem reutilizados em tarefas diversas, de comparar e compreender diferenças entre *workflows* e de redefinir *workflows* por processos baseados em analogia, conforme apresentado na Figura 2.8, onde um *Analogy Template* é empregado para identificar, construir e visualizar a funcionalidade de um determinado processo (*vtkSmoothPolyDataFilter*) no contexto de um experimento modelado no SGWfC Vistrails.

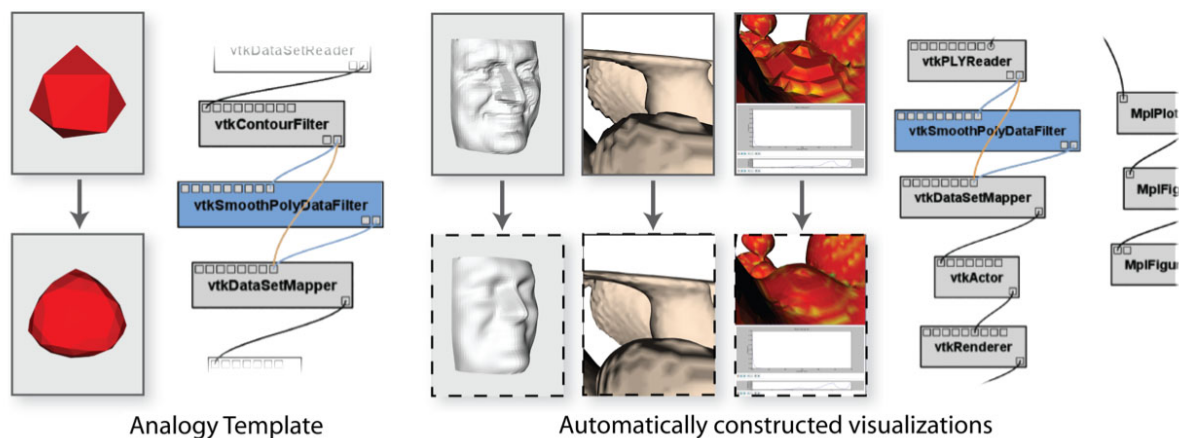


Figura 2.8: Modelos de proveniência em camadas [32]

2.8.1 Modelo para Armazenamento

Pesquisadores têm empregado diversos sistemas de armazenamento para os dados de proveniência. As propostas incluem desde o emprego de arquivos persistidos em dialetos *Extensible Markup Language*³ (XML) e linguagens *web* semânticas até repositórios em SGBD relacionais. Essas soluções são influenciadas por requisitos não-funcionais associados ao escopo do projeto em desenvolvimento, incluindo-se a facilidade de utilização e as restrições impostas pelo ambiente de execução [9].

³XML disponível em www.w3.org/XML/

Um sistema de armazenamento de dados de proveniência baseado em arquivos constitui uma solução mais simples, pois pode ser tratado como um mecanismo nativo do SO subjacente e prescindir de *software* adicional. Por outro lado, outras alternativas como aquelas baseadas em SGBD relacionais em geral requerem processos de instalação e de configuração complementares [10].

Em contrapartida, deve-se ressaltar que a persistência da proveniência a partir de soluções baseadas no modelo relacional provêm um mecanismo de acesso centralizado e eficiente, que permite o acesso compartilhado a diversos grupos de usuários [9].

Barga e Digiampietri [38] defendem que o emprego de um SGBD constitui a alternativa mais adequada para o armazenamento das informações de proveniência. A escolha se sustenta em razão da confiabilidade proporcionada por tal sistema e por oferecer recursos sofisticados para consulta aos dados. Além disso, por se constituir em um entidade desvinculada dos SGWfCs, permite a persistência dos dados de proveniência de forma independente do experimento científico modelado computacionalmente.

Contudo, com a crescente complexidade dos problemas investigados a partir de SGWfC e, conseqüentemente, com a ampliação das informações de proveniência capturadas, torna-se relevante a utilização de técnicas mais eficientes e efetivas para um melhor gerenciamento do volume de dados armazenado, muitas vezes em ambiente distribuído [32].

Existem SGWfCs que utilizam internamente mecanismos de armazenamento da proveniência baseados exclusivamente em SGBDRs, tais como Redux [34] e Taverna [30]. Por outro lado, PASOA [45] e Kepler [13], por exemplo, empregam mecanismos de persistência baseados em arquivos. Vistrails apresenta um mecanismo híbrido para o armazenamento dos dados de proveniência, que utiliza uma base de dados relacional para a proveniência retrospectiva e arquivos XML para a proveniência prospectiva [9].

2.8.2 Modelo para Consulta

Um modelo eficaz de consulta é um componente necessário de um sistema de gerenciamento de dados de proveniência, especialmente quando grandes volumes de informação são registrados.

Dependendo da complexidade do *workflow*, o problema da sobrecarga relacionada à consulta dos dados de proveniência pode ser relevante. Biton *et al.* [8] propõem uma estratégia que emprega abstrações por meio de “visões do usuário” (*user views*). Segundo

esse modelo, o cientista indica quais módulos da especificação do *workflow* são relevantes e o sistema apresenta as informações de proveniência de acordo com as preferências informadas.

Observa-se que essa abordagem adapta-se melhor aos SGWfCs que possuem suporte a abstrações, tais como Vistrails, Taverna e Kepler [9]. Cabe aqui comentar que o modelo de proveniência OPM, adotado pelo presente trabalho, provê suporte a diversos níveis de detalhamento para o coleta e posterior consulta aos dados de linhagem. Essa estratégia adere-se de forma conveniente ao modelo em camadas de abstração proposto em Biton *et al.* [8].

Uma característica comum percebida nas abordagens para a consulta à proveniência refere-se à estreita associação com o modelo de armazenamento empregado [9]. Observa-se a utilização de linguagens de consulta como *Structured Query Language* (SQL), Prolog ou *SPARQL Protocol and RDF Query Language* (SPARQL) ⁴, dependendo da solução de armazenamento adotada [32].

Muito embora essas alternativas sejam adequadas para a consulta a bases de dados, tais linguagens não foram projetadas especificamente para a consulta de proveniência. Como consequência, a sintaxe para a formulação de consultas pode se mostrar difícil e complexa mesmo para a obtenção de resultados triviais.

Porém, mesmo na hipótese de se empregar uma linguagem especificamente projetada para a proveniência, a formulação de consultas a partir de linguagens declarativas baseadas nos metadados do modelo adotado pode constituir-se em uma barreira frente a uma ampla utilização do recurso por parte da comunidade científica.

O mecanismo de consulta à proveniência de dados proposto na arquitetura do *SciProv* utiliza a linguagem SPARQL. Entre as propostas para trabalhos futuros, sugere-se a pesquisa de alternativas baseadas em tecnologias e interfaces mais intuitivas para o cientista. Por exemplo, uma alternativa interessante é implementada pelo Vistrails, que disponibiliza uma interface de consulta gráfica [43]. Essa abordagem oferece aos usuários um mecanismo para a construção de consultas empregando o mesmo ambiente gráfico usado para a concepção de *workflows*, conforme apresentado na Figura 2.9, onde uma consulta visual (*Visual query*) é formulada para verificar a ocorrência de um conjunto de processos no contexto de um experimento científico.

⁴SPARQL disponível em www.w3.org/TR/rdf-sparql-query/

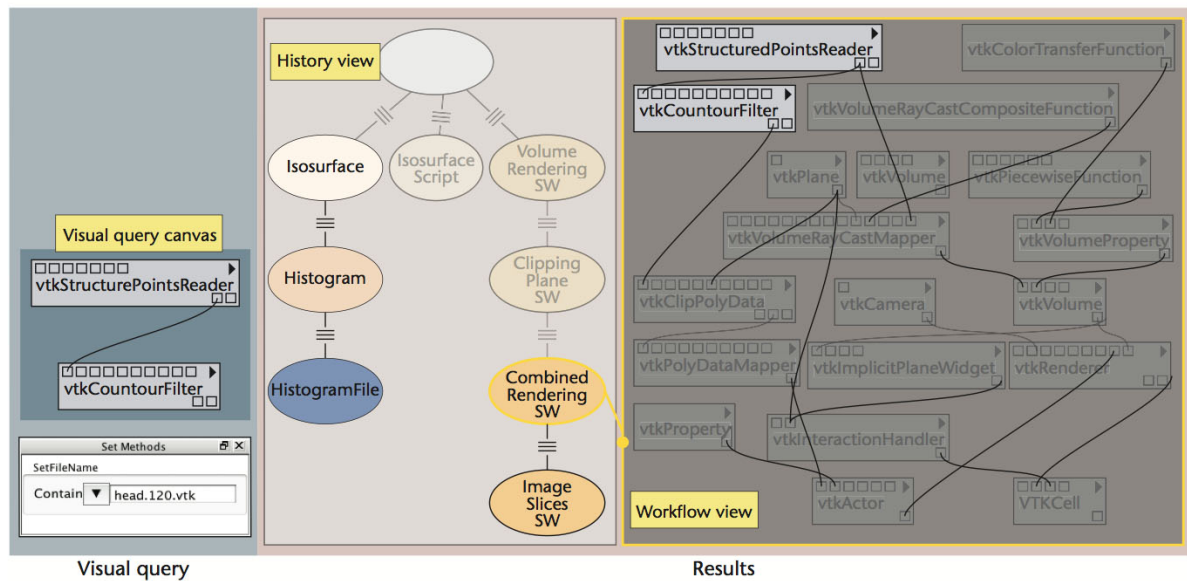


Figura 2.9: Consulta à proveniência de dados a partir de uma interface gráfica [9]

2.8.3 Modelos de Consulta Baseados em Tecnologia Web Semântica

A representação semântica baseia-se no conceito de ontologias para a organização do conteúdo dos dados. Uma ontologia é definida baseando-se no conhecimento de especialistas em um determinado domínio e tem por objetivo elaborar as regras que regulam a combinação entre termos e relacionamentos pertinentes [46]. O emprego de ontologias permite o compartilhamento e a reutilização do conhecimento em um domínio, além de explicitar hipóteses, conforme é o interesse do *SciProv* ao prover recursos para pesquisas em linguagem SPARQL.

No contexto da *web* semântica, as ontologias podem ser explicitadas através de linguagens de marcação que se relacionam segundo o modelo em cascata proposto por Berners-Lee [47], conhecido como “bolo de noiva” (Figura 2.10). Nesse modelo, *Resource Description Framework*⁵ (RDF) e *RDF-Schema* são consideradas as fundações sobre as quais se assentam linguagens de maior abrangência semântica como *Ontology Web Language*⁶ (OWL).

Alguns modelos de proveniência empregam tecnologias provenientes da área de *web* semântica tanto na representação quanto na consulta de dados [49] [50]. Linguagens de marcação como RDF e OWL provêm uma abordagem natural para a proveniência de dados

⁵RDF disponível em www.w3.org/RDF

⁶OWL disponível em http://www.w3.org/2007/OWL/wiki/OWL_Working_Group

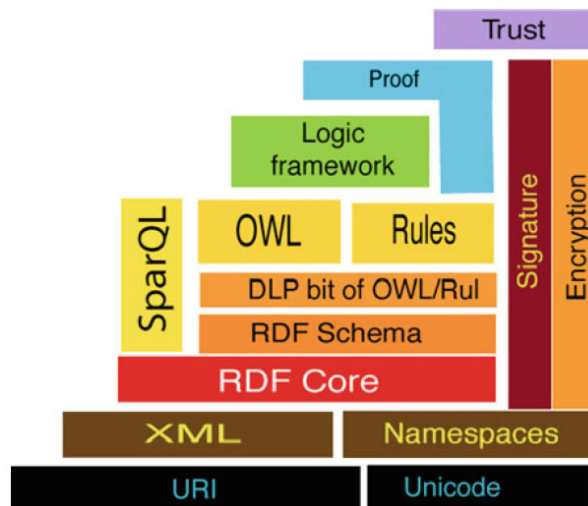


Figura 2.10: Estrutura em camadas da *web* semântica [48]

e apresentam potencial para simplificar a interoperabilidade entre diferentes modelos.

A questão que se impõe refere-se à escalabilidade dessas tecnologias para o tratamento de grandes volumes de dados de proveniência sem impactar a performance de execução dos experimentos científicos [51].

O benefício da abordagem através da *web* semântica consiste na possibilidade de integração de quaisquer fontes de dados na base de conhecimento [49]. Por exemplo, informações armazenadas em uma abordagem relacional podem ser convertidos em dados semi-estruturados — por exemplo, em algum formato de representação de triplas RDF — a partir da especificação de uma ontologia que represente o modelo de proveniência adotado. Mecanismos de serialização de triplas RDF em formato XML são particularmente interessantes nesse contexto.

A infraestrutura de consulta modelada no presente trabalho emprega a estratégia de abordagem acima descrita. Além da possibilidade da integração de informações contidas em diversos repositórios relacionais distribuídos e até mesmo heterogêneos, esse modelo inclui novas alternativas e recursos que podem elaborar consultas ricas e sofisticadas com o emprego de máquinas de inferências e ontologias de domínio. O Capítulo 4 detalha o emprego da tecnologia *web* semântica no contexto do *SciProv*.

2.9 Novas Aplicações para os Dados de Proveniência

A proveniência de dados constitui uma área de estudos atual e que avança rapidamente. Nesse cenário, novos campos de aplicações têm surgido, com alicerces na gerência da proveniência em *workflows*, na habilidade para integração de dados oriundos de diversos sistemas e no desenvolvimento de ferramentas gráficas e analíticas para uma melhor exploração das informações coletadas.

Nesse contexto, surgem os denominados laboratórios científicos colaborativos (*science collaboratories*) [52], com potencial para mudar a forma de se fazer ciência, a partir da disseminação em larga escala de dados de proveniência e também das técnicas e ferramentas concebidas para o tratamento das informações históricas produzidas. O que se vislumbra é que a exploração da proveniência em um ambiente colaborativo possa conduzir os cientistas a um modelo de aprendizagem a partir de exemplos disponíveis, capazes de acelerar as pesquisas em andamento [53].

A área de ensino constitui outra importante aplicação dos SGWfCs com suporte à proveniência, em particular para as disciplinas com ênfase em exploração de dados, tais como mineração de dados e visualização. Nesse contexto, as informações de proveniência podem tornar as aulas mais efetivas e melhorar a experiência de aprendizado por parte dos alunos.

Por exemplo, ao utilizar em sala de aula uma ferramenta com suporte à proveniência, um instrutor pode manter um registro detalhado de todos os passos executados ao longo de um experimento. Estas informações tornam-se assim disponíveis para os estudantes poderem reproduzir a simulação e ampliar o entendimento sobre o assunto abordado [32].

Fundamentalmente, deve-se destacar ainda que o registro detalhado das etapas a serem seguidas para produzir um determinado resultado confere à comunidade científica a possibilidade de reproduzir e validar um experimento.

A partir de 2008, o *Special Interest Group on Management of Data* (SIGMOD) apresentou uma exigência de reprodutibilidade de experimento. O objetivo é especificar um mecanismo sistemático de avaliação da confiabilidade e veracidade dos trabalhos de pesquisa submetidos aos editores científicos. Essa tendência pode ser observada também em publicações diversas como *IEEE Transactions on Signal Processing and Computing in Science and Engineering* (CiSE).

Entretanto, para que iniciativas similares venham a ser largamente adotadas pela

comunidade científica é necessário que se disponha de infraestrutura de gerenciamento de proveniência e ferramental adequado que permita a utilização dos recursos disponíveis por uma ampla faixa de usuários no meio científico [32].

3 OPEN PROVENANCE MODEL

Um modelo de proveniência tem por objetivo especificar o conjunto de informações que são suportadas em uma abordagem de proveniência. Além de prover recursos para a representação da proveniência retrospectiva, o modelo pode fornecer suporte à proveniência prospectiva e a anotações.

O OPM consiste em um esforço por parte de um grupo de pesquisadores envolvidos com o tema da proveniência em prover um modelo que permita a interoperabilidade entre os diversos sistemas de coleta e gerência de linhagem existentes [54]. A versão 1.1, datada de dezembro de 2009, refere-se à mais recente disponível por ocasião do desenvolvimento do presente trabalho.

3.1 Requisitos do Open Provenance Model

A especificação do modelo OPM baseia-se nas seguintes premissas:

1. Permitir o intercâmbio de informações de proveniência entre sistemas a partir de uma camada de compatibilidade baseada em um modelo compartilhado. Pode-se acrescentar que a interoperabilidade constitui a principal característica do modelo OPM.
2. Permitir aos desenvolvedores construir e compartilhar *softwares* capazes de implementar tal modelo de proveniência. Tal processo pode ser alavancado a partir de um conjunto de ferramentas computacionais elaboradas e disponibilizadas por membros da comunidade científica envolvidos na especificação e refinamento do modelo OPM.
3. Permitir a representação da proveniência tanto de processos modelados em computador quanto de processos cuja existência não se relaciona com qualquer tipo de meio digital. Quanto a esta última possibilidade e no intuito de satisfazer a curiosidade do leitor, pode-se pensar nos dados de proveniência referentes a um prosaico bolo caseiro: que ingredientes e utensílios foram usados, quando foi feito, onde, etc.
4. Permitir a coexistência de dados de proveniência em diversos níveis de detalhamento. Essa característica é importante no sentido de permitir a proveniência de dados coletados em uma ampla faixa de granularidade, de acordo com as especificidades

do experimento científico modelado.

A hipótese fundamental do modelo OPM consiste em assumir que a proveniência de qualquer entidade — seja oriunda de um processo executado em um ambiente computacional ou não — pode ser representada a partir de um grafo anotado de causalidade (*annotated causality graph*). Tal estrutura pode ser interpretada como um grafo dirigido acíclico no qual são anexadas anotações adicionais à cerca da execução do processo. Nesse contexto, um grafo de proveniência deve ser entendido como um registro de uma execução passada (ou em andamento) e não como a descrição de algo que poderá ocorrer no futuro [54].

Portanto, o modelo OPM, conforme especificado, tem por objetivo capturar os dados de proveniência retrospectiva e anotações. Em outros termos, o modelo não representa formalmente as informações de proveniência prospectiva a cerca do processo modelado.

Essa característica é, no entanto, coerente com o escopo do projeto, que se caracteriza por uma abordagem para o problema da proveniência independente de qualquer sistema de gerência de dados de linhagem e, em particular, das soluções oferecidas para a questão diretamente a partir dos diversos sistemas de gerência de *workflows* científicos existentes. Em última instância, a proveniência prospectiva pode ser entendida como o próprio *workflow* científico modelado em um SGWfC específico e aderente ao próprio modelo conceitual implementado pela ferramenta computacional.

No entanto, é importante observar que abordagens de proveniência baseadas no modelo OPM podem extrair informações a cerca da proveniência prospectiva seja diretamente a partir das conexões entre os componentes do grafo de causalidade ou a partir de inferências, utilizando-se, por exemplo, recursos de *web* semântica, como ontologias e máquinas de inferência.

A abordagem de proveniência descrita no presente trabalho segue essa proposta, no sentido de prover recursos sofisticados de consulta às informações de proveniência coletadas.

3.2 Descrição do Modelo OPM

O modelo OPM permite caracterizar as causas que deram origem a uma informação de proveniência. Em essência, consiste em um grafo dirigido que expressa os relacionamento

de dependência que originaram um dado específico. Os elementos constituintes desse grafo dirigido são apresentados a seguir.

3.2.1 *Artefato, Processo e Agente*

O principal objetivo do modelo OPM é permitir a representação de como uma informação chegou em um dado instante de tempo a um determinado estado com um conjunto específico de características. O modelo deve ser capaz de representar a linhagem tanto de entidades em ambientes computacionais — como em *workflows* científicos —, como também em processos executados manualmente — *in-vivo* e *in-vitro*, por exemplo. De fato, mesmo uma prosaica sequência de passos executados na preparação de um bolo caseiro pode ser modelado através de um grafo de proveniência OPM.

O modelo de proveniência formula o conceito de *artefato* como uma entidade cujo estado é imutável. Do mesmo modo, formula o conceito de *processo* como sendo uma ação que resulta em um ou mais artefatos.

Um processo normalmente ocorre em um contexto que dê início, dispare ou facilite a sua execução. Exemplos de tais contextos são variados e podem incluir um local específico onde o processo é executado, uma pessoa que tenha controle sobre o processo ou uma instituição patrocinadora do processo. Estas entidades são definidas como *agentes* e podem ser entendidos como catalisadores de um processo em curso.

Assim, o modelo OPM fundamenta-se em três entidades principais, definidas formalmente conforme se segue:

- *Artefato* – uma entidade cujo estado é imutável, que pode ter representação física no mundo real ou representação computacional em um sistema informatizado.
- *Processo* – ação ou conjunto de ações realizadas em artefatos ou causadas por artefatos que resultam em novos artefatos.
- *Agente* – entidade contextual que age como um processo catalizador, habilitando, facilitando, controlando e afetando a execução.

O OPM é um modelo que tem por objetivo descrever artefatos no passado, ou seja, procura explicar como foram derivados. Da mesma forma, os processos também ocorrem no passado, isto é, eles já terminaram a execução — ou os processos estão ainda em execução. Portanto, o OPM não se destina a descrever possíveis estados futuros de artefatos

bem como as atividades de processos ainda não ocorridos. Em outros termos, o modelo proposto pelo OPM visa capturar a proveniência retrospectiva e anotações.

Para facilitar a compreensão e promover uma representação visual compartilhada, o modelo OPM apresenta uma notação gráfica e uma definição formal para os grafos de proveniência, onde os artefatos são representados por círculos, os processos são representados por retângulos e os agentes são representados por octágonos (Figura 3.1).

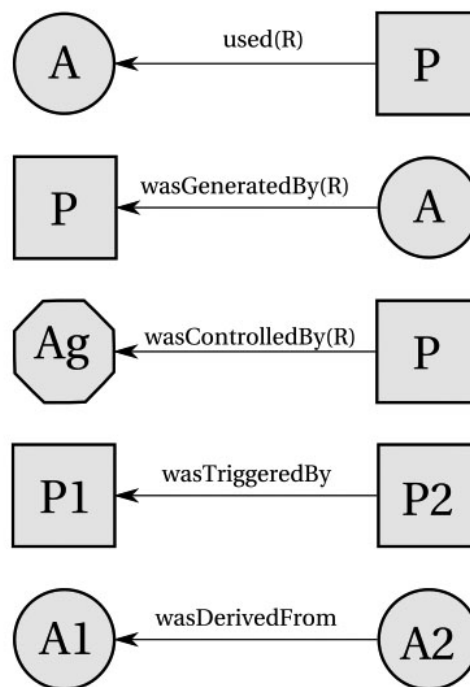


Figura 3.1: Entidades e dependências no modelo OPM [54]

3.2.2 Dependências Causais

O modelo OPM visa capturar as *dependências causais* entre artefatos, processos e agentes. Portanto, um grafo de proveniência pode ser definido como um grafo dirigido cujos nós são artefatos, processos e agentes e cujas arestas pertencem a uma das seguintes categorias representadas na Figura 3.1. Uma aresta representa uma dependência causal entre a origem — que denota o efeito — e o destino — que indica a causa.

As duas primeiras especificações apresentadas na Figura 3.1 expressam, respectivamente, que um processo “usou” (*used*) um artefato e que um artefato “foi gerado por” (*was generated by*) um processo.

3.2.2.1 Papéis

Uma vez que um processo pode ter usado diversos artefatos, é importante identificar os papéis (*roles*) — identificados pela letra R — nos quais esses artefatos foram usados. Da mesma forma, um processo pode ter gerado muitos artefatos, onde cada um poderia ter um papel específico. Por exemplo, uma operação de divisão usa dois operandos, que assumem os papéis de dividendo e divisor e gera dois números como resultado, que assumem os papéis de quociente e resto.

É importante mencionar que os papéis são relevantes somente no âmbito do processo onde são definidos e que o significado específico de cada papel não é definido pelo modelo OPM, mas pelo domínio da aplicação. Essa característica é interessante porque confere a uma arquitetura que implemente o modelo OPM a possibilidade de definir formalmente os papéis utilizados em um domínio específico a partir de uma ontologia e, a partir dessa infraestrutura, utilizar máquinas de inferência para processar consultas com base no conhecimento explicitado.

3.2.2.2 Dependência Causal *was controlled by*

Um processo é iniciado por um agente, que atua como um catalisador ou controlador. Esta dependência causal é expressa por uma aresta do tipo “foi controlado por” (*was controlled by*). Levando-se em conta que um processo pode ter sido controlado por vários agentes, torna-se necessário identificar os papéis dos controladores. Observe que a dependência entre um agente e um processo representa uma relação de controle, e não uma relação de derivação de dados.

3.2.2.3 Dependência Causal *was derived from*

Mesmo que um artefato A_2 tenha sido gerado a partir de um processo que usou um ou mais artefatos, isto não especifica qual ou quais artefatos são necessários para gerar A_2 . Assim, para tornar essa dependência causal explícita, é necessário afirmar que o artefato A_2 “foi derivado de” (*was derived from*) outro artefato A_1 . Essa dependência confere uma visão orientada para o fluxo de dados de proveniência no modelo OPM.

3.2.2.4 Dependência Causal *was triggered by*

Há situações em que não é possível saber exatamente que artefato foi usado por um processo P_2 , mas que houve algum artefato gerado por outro processo P_1 . O processo P_2 é dito, nesse caso, ter “sido disparado por” (*was triggered by*) P_1 . Em contraste com a aresta *was derived from*, uma aresta *was triggered by* permite a adoção de uma visão orientada a processos de execuções passadas.

3.2.2.5 Convenção para Relações de Causalidade

Respeitando-se as convenções usadas pelo modelo OPM, observa-se que as relações de causalidade apresentadas usam o pretérito para indicar que se referem à execuções passadas. Esses relacionamentos de causalidade são definidos como se segue:

Uma relação de causalidade é representada por um arco e denota a presença de uma dependência causal entre a origem do arco (o efeito) e o destino do arco (a causa).

Conforme apresentado na Figure 3.1, cinco relacionamentos de causalidade são reconhecidos no modelo OPM:

- Um processo usou um artefato.
- Um artefato foi gerado por um processo.
- Um processo foi disparado por um outro processo.
- Um artefato foi derivado de um outro artefato.
- Um processo foi controlado por um agente.

Por meio de anotações é possível ainda que as arestas possam se dividir em subtipos destas cinco categorias.

3.2.2.6 Dependências Causais Múltiplas

Noções de dependências causais múltiplas foram avaliadas na elaboração do modelo OPM. Nesse contexto, pode-se argumentar que uma noção muito forte de dependência causal poderia expressar que um conjunto de entidades seria *necessário e suficiente* para determinar a existência de outra entidade. Considerou-se que tal noção não seria adequada, pois, segundo o pressuposto de mundo aberto (*open world assumption*), sempre se poderia argumentar que outros fatores ainda poderiam ter influenciado em um resultado.

Portanto, na elaboração do modelo OPM, considerou-se que noções mais fracas, que expressem dependências necessárias, seriam mais adequadas. No entanto, mesmo assim, é possível distinguir as dependências de dados — por exemplo, quando um quociente depende do dividendo e do divisor — de uma dependência de controle onde a presença de algum artefato ou o início de um processo pode determinar a presença de uma outra entidade.

Por conseguinte, houve uma série de fatores que influenciaram os desenvolvedores do modelo a adotar uma noção fraca de dependência causal para o OPM:

- *Expressibilidade* – Espera-se que os sistemas sejam capazes de descrever o que os componentes estão processando, sem que seja necessário um profundo conhecimento dos dados e dependências de controle internos. Noções fracas de dependência são importantes para que tais sistemas possam ser capazes de usar o OPM na prática.
- *Modularidade* – O OPM suporta o detalhamento da proveniência em multiníveis. Em um grafo de causalidade que consiste na composição paralela em diversos níveis de detalhe, o subgrafo de mais alto nível requer uma noção mais fraca de dependência em relação ao subgrafo de mais baixo nível.

3.2.2.7 Dependências Causais Suportadas

Portanto, com base nos pressupostos apresentados, o modelo OPM adota as dependências causais descritas a seguir. Deve-se observar, porém, que subclasses destas dependências, capazes de capturar dependências mais fortes de causalidade, podem ser definidas em sistemas específicos, e futuramente, vir a ser incorporadas no modelo OPM.

Novamente, têm-se aqui uma característica interessante do modelo OPM. Confere-se a sistemas específicos de proveniência que adotem o OPM a possibilidade de definir e implementar novas relações de dependências causais que sejam aderentes ao domínio tratado. Por exemplo, pode-se considerar relevante uma dependência causal do tipo *was derived from* mais forte no contexto da coleta e gerência de *workflows* científicos em um determinado domínio estudado.

O protótipo implementado para o sistema de proveniência proposto trata a questão a partir da extensão da ontologia que descreve o modelo OPM, necessária para o processamento de inferências em consultas SPARQL ao grafo de linhagem. O capítulo 4 descreve em detalhes o enfoque adotado no *SciProv* para tirar proveito dessa possibilidade.

- *Artefato usado por um processo* – uma aresta do tipo *used* que parte de um processo para um artefato consiste em um relacionamento causal cujo objetivo é indicar que o processo requer a disponibilidade de um artefato para ser capaz de completar a execução. Quando diversos artefatos estão conectados a um mesmo processo a partir de múltiplas arestas *used*, implica dizer que todos os artefatos são necessários para que o processo se complete.

Em uma interpretação mais forte da aresta *used*, pode-se considerar que o artefato deva estar disponível para que o processo seja iniciado. Conforme mencionado, o OPM considera que essa noção possa ser validada em determinadas circunstâncias e, por conseguinte, permitir que seja definida como um subtipo de *used*.

Vale mencionar que uma interpretação mais rígida não pode ser considerada composicional, pois um artefato A pode ser requerido para iniciar a execução de um processo P_1 , mas isso não implica que A seja necessário para dar início a um processo P_2 que seja um super-processo de P_1 .

- *Artefatos gerados por processos* – uma aresta do tipo *was generated by* que parte de um artefato para um processo consiste em um relacionamento causal cujo objetivo é indicar que a geração de um artefato requer que um processo seja iniciado. Quando diversos artefatos encontram-se conectados a um mesmo processo através de múltiplas arestas do tipo *was generated by*, significa que o processo deve ter início para que todos os artefatos possam ser gerados.

De forma análoga, uma interpretação mais forte consiste em considerar que o processo seja completado para que o artefato seja gerado. Essa alternativa não é admitida pelo OPM em razão de impor severas restrições à modelagem da troca de artefatos entre processos em execução.

- *Processo disparado por processo* – uma aresta do tipo *was triggered by* que parte de um processo P_2 para um processo P_1 consiste em uma dependência causal cujo objetivo é indicar que o início do processo P_1 é necessário para que o processo P_2 possa ser completado.

Deve-se observar que as dependências causais anteriormente descritas são expressas somente como condições necessárias. Essa interpretação é mais fraca do que a interpre-

tação usual para *disparar*, que tende a expressar uma condição suficiente para que um evento seja realizado.

- *Artefato derivado de artefato* – uma aresta do tipo *was derived from* que parte de um artefato A_2 para um artefato A_1 consiste em um relacionamento causal cujo objetivo é indicar que o artefato A_1 requer que seja gerado a partir do artefato A_2 .
- *Processo controlado por agente* – uma aresta do tipo *was controlled by* que parte de um processo P para um agente Ag consiste em uma dependência causal que indica que o início e o término de um processo P foi controlado por um agente Ag .

3.2.3 *Objetivo dos Papéis no Modelo OPM*

Os papéis (*roles*) são associados às arestas *used*, *was generated by* e *was controlled by*. A função de um *role* consiste em fazer a distinção entre os tipos de dependência existentes quando múltiplas arestas estão conectadas a um mesmo processo.

- *Papel* – um papel (*role*) especifica a função de um artefato ou de um agente em um processo.

Um papel é usado para a fazer a distinção entre os diversos relacionamentos de uso, geração e controle possíveis no modelo OPM.

1. Um processo pode usar mais de um artefato, possivelmente para diferentes propósitos. Cada relação do tipo *used* pode ser determinada por um papel específico no contexto do processo. Por exemplo, um processo pode utilizar dois arquivos, ao ler parâmetros de um (*role = parameters*) e ler dados de outro (*role = data*).
2. Um artefato pode ser usado por mais de um processo, possivelmente com propósitos distintos. Neste caso, as relações do tipo *used* podem ser diferenciadas entre si através dos respectivos papéis. Por exemplo, um dicionário poderia ser usado por um processo para verificar a grafia de proveniência (*role = look up provenance*) enquanto outro processo poderia usar o mesmo dicionário para verificar a grafia de outra palavra.
3. Um agente pode controlar mais de um processo. Nesse caso, os diferentes processos podem ser diferenciados através do papel associado com as relações do tipo *was controlled by*.

4. Um processo pode ser controlado por mais de um agente. Nesse caso, cada agente deve possuir uma função de controle diferente, que pode ser determinado a partir dos papéis associados aos relacionamentos do tipo *was controlled by*.

Segundo a perspectiva do modelo OPM, os papéis têm uma natureza unicamente sintática e o escopo limita-se aos processos e agentes aos quais encontram-se relacionados. Em outras palavras, um papel somente possui um significado no contexto de um dado processo ou agente específico. Para um dado processo, cada relacionamento do tipo *used*, *was generated by* ou *was controlled by* pode ter um papel específico. Tais papéis, porém, não têm significado no contexto de outro processo. Deve-se observar que o modelo OPM não impõe a exclusividade de papéis no contexto de um processo. Por exemplo, ao fazer um bolo com dois ovos pode-se definir cada ovo como um artefato separado e as duas arestas usariam um mesmo papel (ovo).

Novamente, cabe ressaltar a possibilidade de agregar conteúdo semântico aos papéis, conforme modelados no OPM. Tal recurso pode permitir a formulação de consultas aos metadados de proveniência com base no significado semântico dos papéis no contexto de um domínio estudado.

O próprio documento que formaliza o modelo OPM vislumbra essa possibilidade. Segundo Moreau *et al.* [54] um aspecto importante a ser considerado refere-se à especificação dos papéis. No sentido de prover uma melhor interoperabilidade, os autores consideram que seria desejável estabelecer-se um conjunto de papéis-padrão cujo significado fosse acordado no âmbito da comunidade de usuários do OPM.

3.2.4 Exemplos

Um exemplo que ilustra as entidades e algumas dependências causais é apresentado na Figura 3.2. Nesse exemplo, é exibido um subconjunto dos metadados de proveniência referente a uma saída *Atlas × Gráfico* gerado a partir da execução de um *workflow* durante o *First Provenance Challenge*¹.

O usuário responsável pelo controle do processo foi *John Doe*. Arestas do tipo *used*, *was generated by* e *was controlled by* estão representadas por linhas pontilhadas e os papéis estão descritos entre parênteses. A derivação dos dados está representada através de arestas do tipo *was derived from* (linhas mais grossas). Deve-se observar que o fato de

¹Disponível em <<http://twiki.ipaw.info/bin/view/Challenge/FirstProvenanceChallenge>>

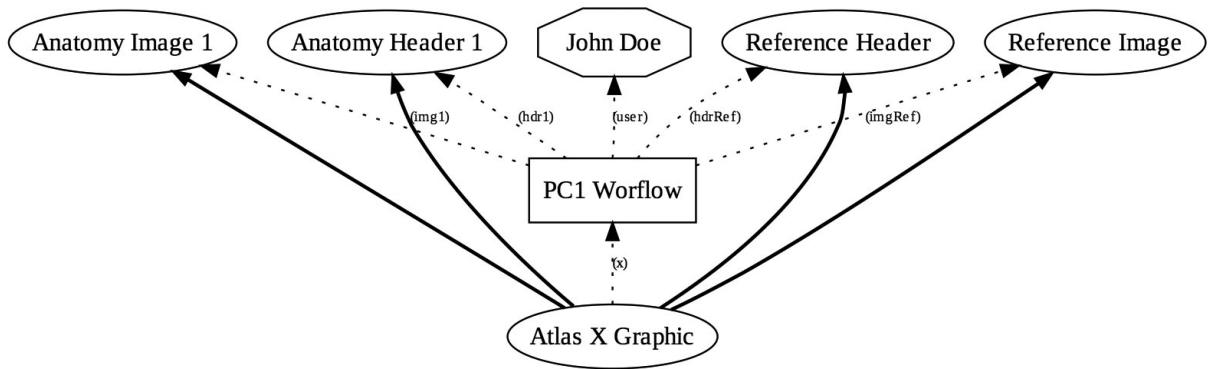


Figura 3.2: Exemplo de grafo de proveniência segundo o modelo OPM [54]

um processo ter usado um artefato e gerado outro artefato não implica que este último seja derivado da anterior. No modelo OPM, essa relação tem de ser definida de forma explícita.

É importante frisar que a concepção do modelo OPM não se limita a artefatos e processos manipulados em um ambiente computacional. Na Figura 3.3, um grafo de proveniência expressa que João assou um bolo preparado a partir dos ingredientes manteiga, ovos, açúcar e farinha.

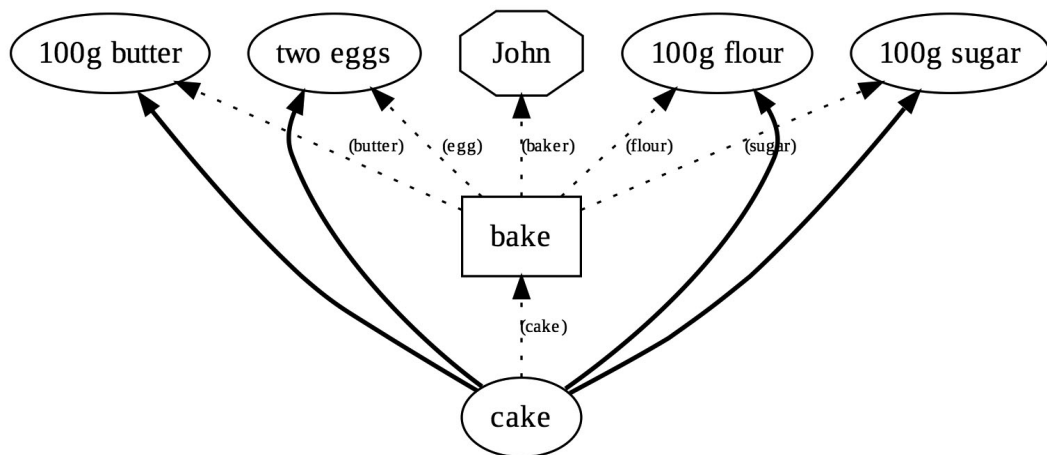


Figura 3.3: Grafo de proveniência “Bolo Assado pelo João” [54]

Pelos exemplos apresentados, é possível observar que os grafos podem ser construídos de forma incremental a partir da interconexão de artefatos, processos e agentes através de arestas. O significado das relações de causalidade pode ser entendido no contexto das arestas *used* ou *was generated by* utilizadas em cada processo. A partir da conexão de um processo a vários artefatos através de arestas do tipo *used*, o modelo OPM não só expressa as entradas (artefatos) para o processo em questão como também a dependência causal

com esses artefatos. Assim, o processo somente pode ser executado se todos os artefatos relacionados estiverem disponíveis.

Da mesma forma, quando se expressa que diversos artefatos foram gerados por um processo, isso implica em dizer que esses artefatos não teriam existido se o processo não tivesse começado a sua execução. Além disso, todos os artefatos foram gerados pelo processo, ou seja, um não poderia ter sido gerado sem que os outros também o fossem. A implicação é que a geração de qualquer artefato é causada pelo processo que, para sua execução, necessita da disponibilidade de todos os artefatos requeridos.

Como ilustrado nos exemplos apresentados na Figura 3.2 e na Figura 3.3, as entidades e as arestas definidas até esse ponto permitem captar muitos dos casos de uso encontrados na literatura de proveniência [54]. No entanto, não permitem fornecer descrições em níveis múltiplos de abstrações ou a partir de pontos de vista diferentes. Para apoiar estes casos, torna-se necessário que múltiplas descrições de uma mesma execução possam coexistir.

3.3 Descrições Sobrepostas e Hierárquicas

A Figura 3.4 mostra dois exemplos de grafos de proveniência que descrevem a história progressiva da lista (3,7). De acordo com o lado esquerdo do grafo, a lista foi gerada por um processo que adicionou uma unidade a todos os componentes da lista (2,6). De acordo com o lado direito do grafo, o processo de derivação de (3,7) exigiu que a lista fosse criada a partir dos valores 3 e 7, respectivamente, obtidos pela adição de uma unidade aos valores 2 e 6. Esses valores, por sua vez, foram obtidos através do acesso ao conteúdo da lista original (2,6). Para facilitar a compreensão dessas figuras, arestas do tipo *was derived from* foram anotadas. As anotações foram explicitadas utilizando-se rótulos anexados às arestas.

Assumindo-se que ambos os grafos referem-se às mesmas listas (2,6) e (3,7), pode-se observar que representam a proveniência capturada em diferentes níveis de detalhamento. A exigência de se fornecer detalhes em diferentes níveis de abstração ou a partir de diferentes pontos de vista é comum para os sistemas de proveniência e, portanto, espera-se que possam ainda ser integrados em um único grafo.

Na Figura 3.5 é possível verificar como os dois grafos de proveniência da Figura 3.4 foram integrados a partir da escolha de cores diferentes para os nós e para as arestas. O

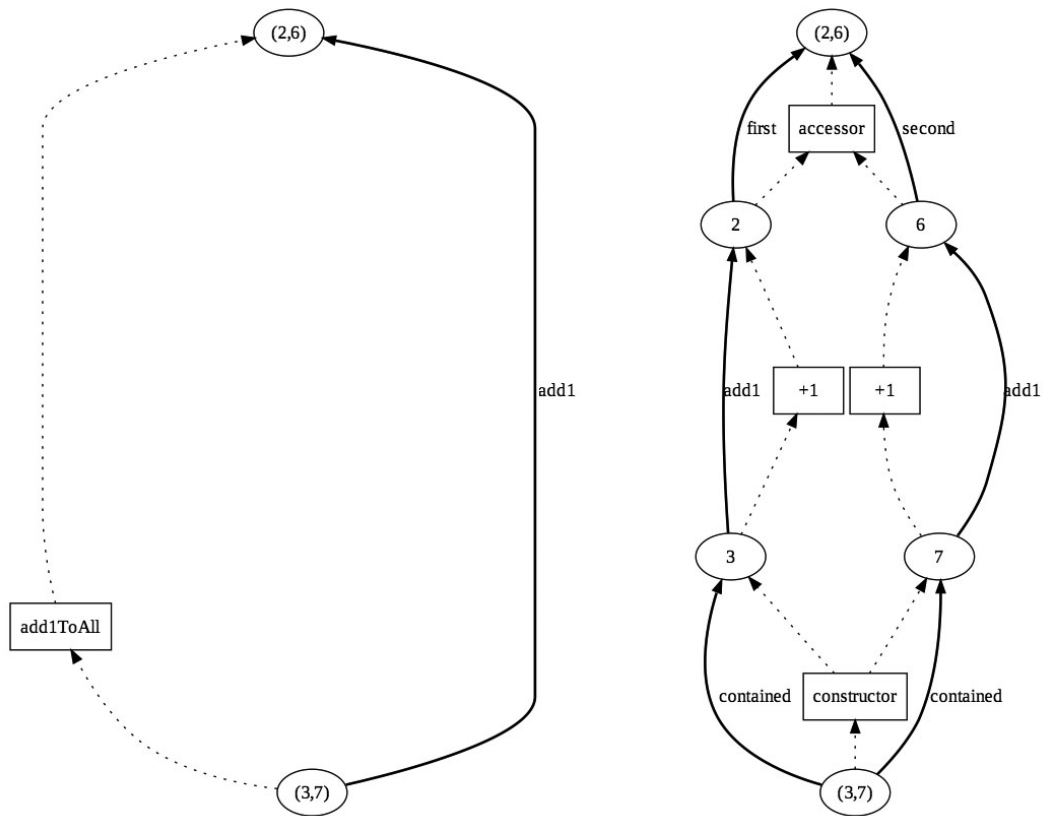


Figura 3.4: Grafos de proveniência em diferentes níveis de detalhamento [54]

desenho em cinza claro refere-se ao lado esquerdo do grafo da Figura 3.4 enquanto que o desenho em preto refere-se à descrição alternativa do grafo direito da Figura 3.4.

Os subgrafos de proveniência em cores distintas consistem em duas descrições distintas porém sobrepostas da execução de um mesmo processo e que oferecem diferentes níveis de detalhamento. Tais subgrafos recebem o nome de “descrições sobrepostas” (*overlapping accounts*) porque compartilham os nós (2,6) e (3,7). Além disso, o subgrafo representado em preto fornece mais detalhes do que o subgrafo representado em cinza claro. Disse-se que o subgrafo preto se constitui em um refinamento do subgrafo em cinza claro.

Ao observar-se a Figura 3.5, é interessante contrastar as arestas do tipo *was generated by* provenientes do artefato (3,7) com as arestas do tipo *used* originadas a partir do processo *constructor*.

Segundo o grafo de proveniência, as arestas do tipo *used* que partem do processo *constructor* significam que ambos os artefatos 3 e 7 foram necessários para que o processo fosse executado. Por outro lado, uma vez que as arestas *was generated by* do artefato (3,7) estão em uma cor diferente, indicam que existem explicações alternativas para o processo que gerou esse artefato.

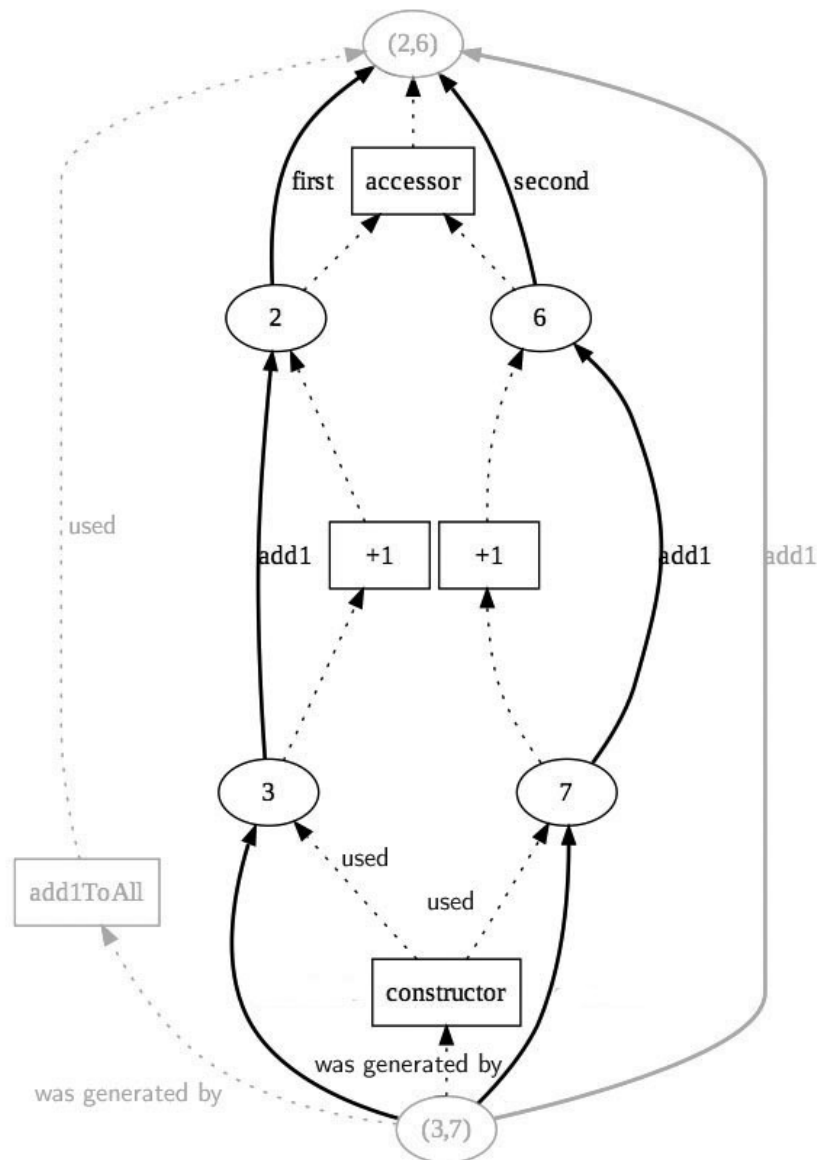


Figura 3.5: Descrições sobrepostas e hierárquicas em um grafo de proveniência [54]

É possível usar refinamentos sucessivos para criar uma hierarquia de descrições, como ilustrado na Figura 3.6. Nesse exemplo, uma terceira descrição, em cinza médio, é introduzida para explicar como um dos processos `+1` foi executado.

Através da combinação de várias descrições, é possível obter ciclos, conforme apresentado no lado esquerdo da Figura 3.7. No lado esquerdo desse exemplo, na primeira descrição (em preto) é apresentado o detalhamento de dois processos $p1a$ e $p1b$ e as respectivas dependências — artefatos $A0$, $A1$, $A2$ e $A3$. Na segunda descrição (em cinza claro) é declarado que os processos $p1a$ e $p1b$ constituem um único processo executado a partir das entradas $A0$ e $A2$ e que produz os artefatos $A1$ e $A3$.

Ao combinar-se as duas visões, um ciclo de arestas do tipo *used* e *was generated by*

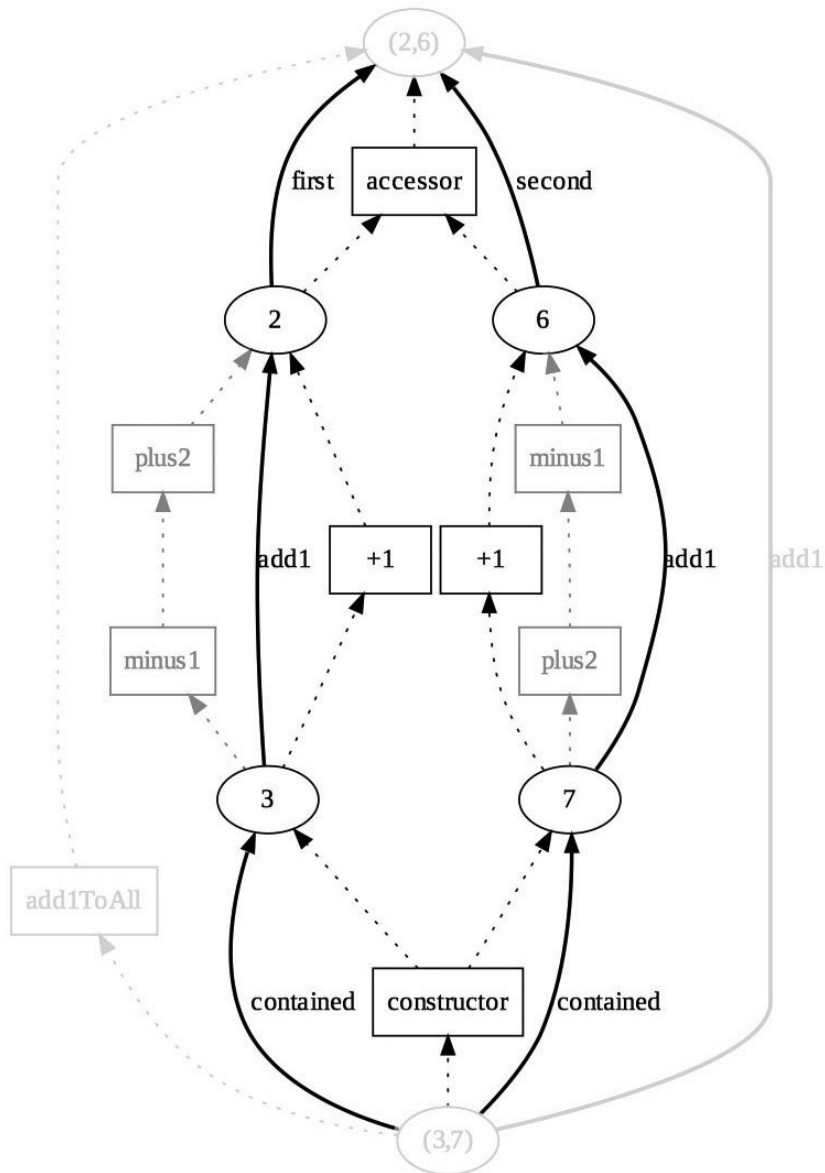


Figura 3.6: Refinamentos sucessivos em um grafo de proveniência [54]

foi criado: $A2 \rightarrow P2 \rightarrow A1 \rightarrow P1 \rightarrow A2$. No lado direito da Figura 3.7 o grafo de causalidade contém derivações explícitas de dados. Neste exemplo, pode-se observar que nenhum ciclo de arestas do tipo *was derived from* é criado, uma vez que as duas descrições se correlacionam — uma delas fornece maior detalhamento do que a outra. No caso mais geral, onde as descrições podem ser distintas, é possível antecipar um ciclo de arestas do tipo *was derived from* resultante da união de múltiplas descrições [54].

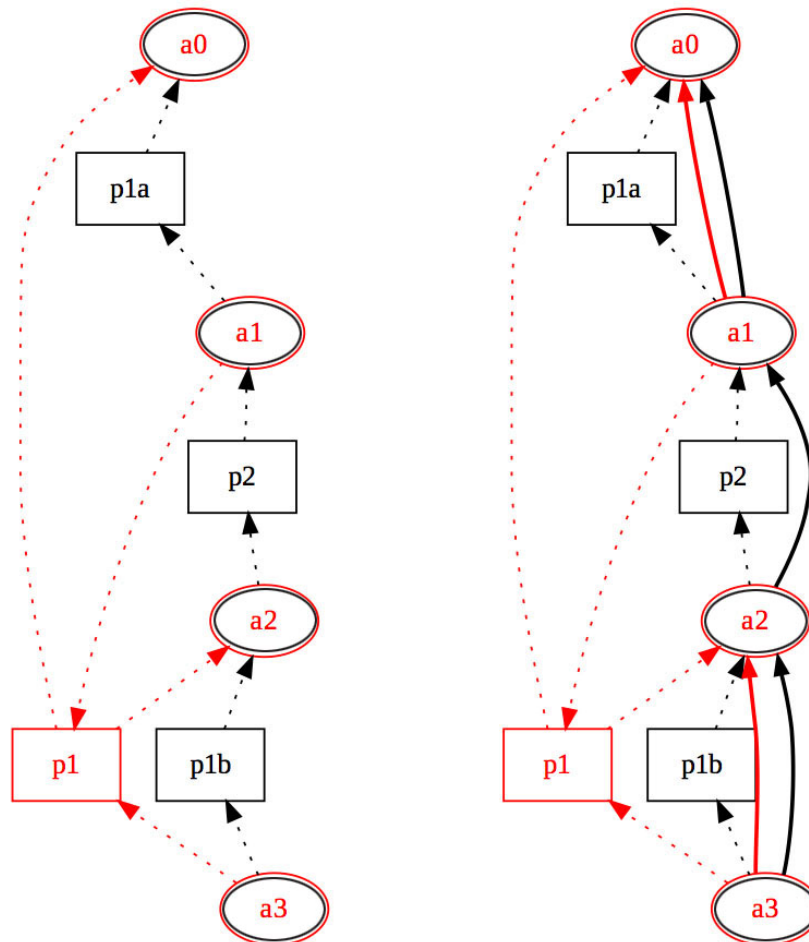


Figura 3.7: Composição de um ciclo a partir de múltiplas descrições [54]

3.4 Restrições Temporais e Observação do Tempo

O modelo OPM permite que informações relacionadas ao controle do tempo sejam incorporadas aos grafos de causalidade. Entretanto, o controle do tempo não se destina a prover derivações de causalidade. Como já apresentado anteriormente, dependências causais precisam ser explicitadas através de arestas entre as entidades do grafo de proveniência.

No modelo OPM, considera-se que se um mesmo “cronômetro” é usado para medir o tempo transcorrido, tanto para o efeito quanto para a causa. Então, o instante do efeito deve ser superior ao instante da respectiva causa. Assim, o tempo pode ser útil para validar derivações de causalidade.

Além disso, o tempo pode ser associado a “ocorrências instantâneas” em um processo. OPM reconhece quatro tipos de ocorrências instantâneas, que podem ser compreendidos tanto em situações relacionadas ao mundo real quanto a sistemas informatizados. Dois

tipos de ocorrências instantâneas estão relacionados a artefatos e dois outros tipos dizem respeito a processos. Para artefatos, é possível considerar as ocorrências de “criação” e de “utilização”, enquanto que para processos, consideram-se o “início” e o “término”.

A justificativa para a adoção do conceito de “tempo instantâneo” no contexto do modelo OPM pode ser considerada equivalente àquela adotada para a hipótese dos artefatos serem entidades de estado imutável [54].

Em alguns cenários, porém, as ocorrências de criação ou de utilização de artefatos e as situações de início ou de término de processos não podem ser consideradas instantâneas. Em tais situações, o detalhamento dos processos e artefatos e das respectivas dependências de causalidade precisam ser explicitadas para que possam ser expressivas no modelo OPM. Por exemplo, o início de funcionamento de uma usina nuclear não pode ser adequadamente modelado como sendo uma ocorrência instantânea quando se tenta entender as falhas que ocorreram durante esta atividade.

No modelo OPM, espera-se que as informações relacionadas ao tempo sejam registradas por um observador de posse de um cronômetro durante uma ocorrência. Deve-se observar aqui que o OPM considera todos os cronômetros sincronizados.

Dado que o tempo é observado, a precisão é estabelecida pela granularidade do cronômetro e pela granularidade das atividades do observador. Assim, enquanto a noção de tempo considerada é instantânea, o modelo admite um intervalo de precisão de acordo com a granularidade dos cronômetro e dos observadores. Assim, no modelo OPM, uma ocorrência instantânea que ocorre em um momento t é especificada a partir de duas observações de tempo T^m e T^M . Desse modo, uma ocorrência instantânea é determinada por ter ocorrido no mais tardar em T^M e não antes de T^m . Assim, $t \in [t^m, T^M]$.

De forma concreta, para um artefato é possível afirmar que foi usado (ou gerado por) não antes que um tempo t_1 nem depois que um tempo t_2 . De forma semelhante, para um processo é possível afirmar que iniciou (ou terminou) não antes que um tempo t_1 nem depois que um tempo t_2 .

Na Figura 3.8 são rerepresentadas as entidades do OPM incluindo as informações de tempo que podem ser expressas no modelo. É importante destacar que as informações de tempo são opcionais no OPM e que são expressas na forma de um intervalo de tempo entre duas observações.

Arestas do tipo *used* e *was generated by* podem receber um rótulo opcional, indicando

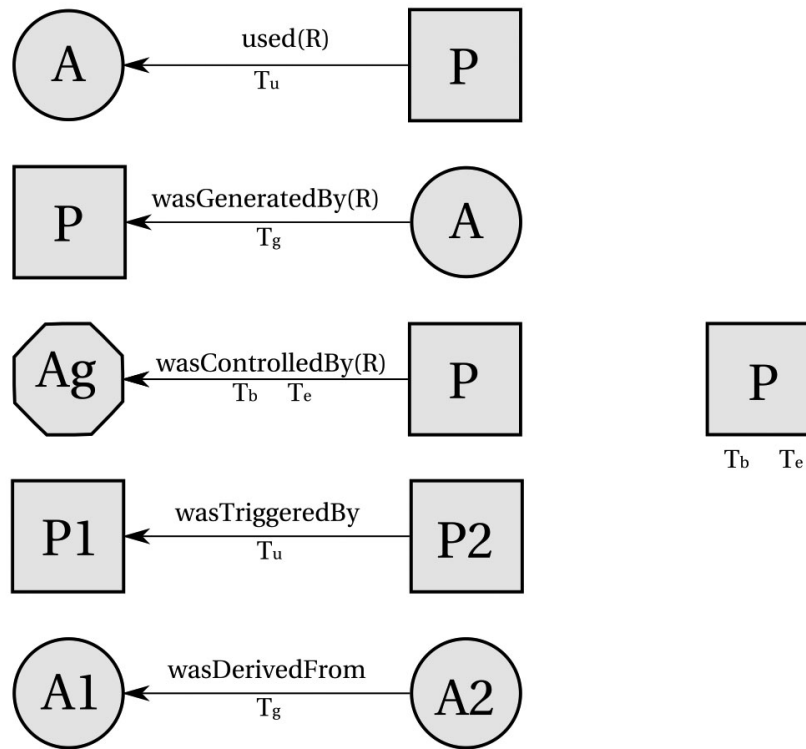


Figura 3.8: Tempo no modelo OPM [54]

que o artefato associado foi usado ou gerado em um dado momento conhecido T_u ou T_g respectivamente. Para uma aresta do tipo *was controlled by*, o modelo admite dois rótulos opcionais que representam o tempo do início T_b e do término T_e da execução de um processo. Para uma aresta do tipo *was derived from*, um rótulo opcional é permitido para indicar quando o artefato foi gerado (T_g). Finalmente, para uma aresta do tipo *was triggered by*, o modelo admite um rótulo opcional T_u para indicar quando o processo foi usado.

Ressalta-se, porém, que o modelo de causalidade em OPM é essencialmente atemporal uma vez que a ordem de precedência não implica em causalidade. Se um processo $P1$ foi executado antes que um processo $P2$ não se pode inferir que $P1$ causou a execução do processo $P2$ [54].

A Figura 3.9 exibe vários relacionamentos do tipo *ocorreu antes* que são admitidos pelo modelo OPM. A notação $T_1 \leq T_2$ expressa que um evento observado no tempo T_1 ocorreu anteriormente ao evento observado no tempo T_2 . Quando as duas observações de tempo são feitas com o mesmo cronômetro — ou os cronômetros encontram-se sincronizados —, implica que podem ser comparadas.

De acordo com a Figura 3.9, um artefato deve existir antes de ser usado ($T_1 \leq T_3$

e $T_4 \leq T_6$). Se um artefato é usado por um processo, esse evento deve ocorrer após o início da execução do processo ($T_2 \leq T_3$). Um processo deve gerar um artefato antes de terminar a execução ($T_4 \leq T_5$). Além disso, o início do processo deve preceder a geração do artefato ($T_2 \leq T_4$) e deve terminar depois da geração do artefato ($T_2 \leq T_5$).

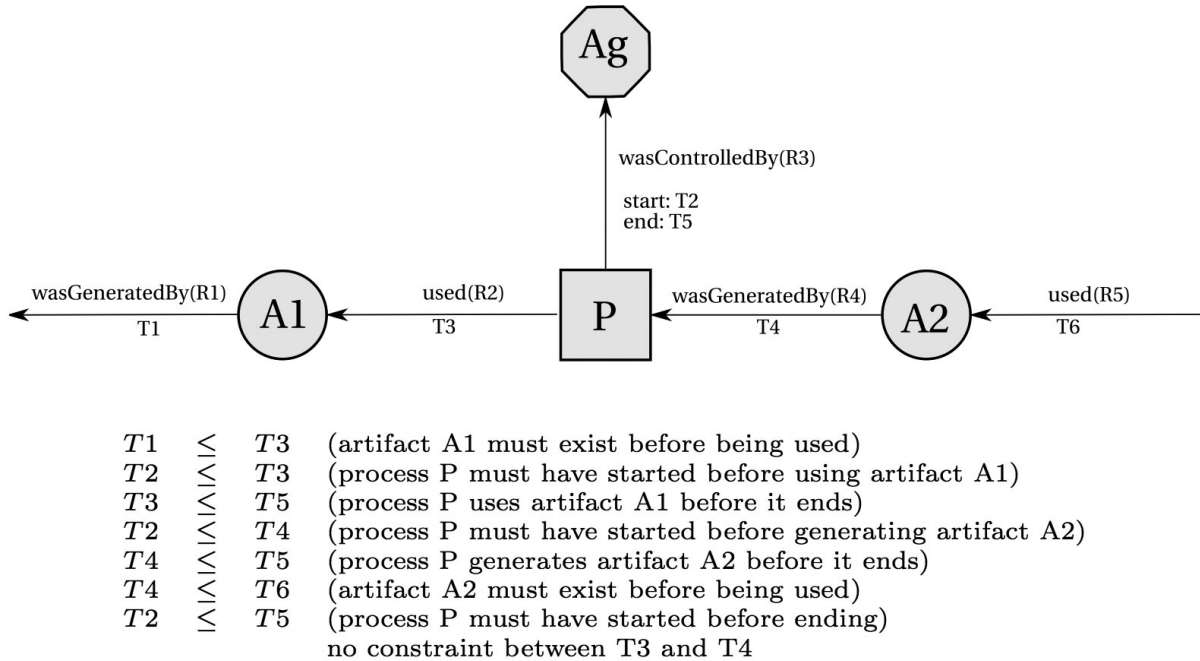


Figura 3.9: Relacionamentos do tipo *ocorreu antes* no modelo OPM [54]

3.5 Completude e Inferências

A noção de grafo OPM foi estabelecida com base em um conjunto de regras sintáticas e restrições topológicas. Grafos de proveniência têm por objetivo representar dependências de causalidade capazes de explicar a história progressiva de processos e artefatos. Nesse contexto, espera-se que diversos algoritmos de inferência possam explorar esse modelo de dados a fim de prover novas e poderosas funcionalidades aos usuários.

3.5.1 Regras de Completude

A Figura 3.10 e a Figura 3.11 descrevem as chamadas “regras de completude” (*completion rules*), que se constituem nas inferências que podem ser processadas a partir do modelo OPM. Uma regra explica como um subgrafo pode ser convertido em outro subgrafo.

A Figura 3.10 exibe uma transformação bidirecional, em outras palavras, uma equivalência. De acordo com a transformação denominada “eliminação de artefato”, uma aresta do tipo *was triggered by* é inferida a partir da ocorrência de arestas do tipo *used* e *was generated by*. A Figura 3.10 mostra que esta regra de conclusão é bidirecional.

A transformação chamada “introdução de um artefato” permite estabelecer que a aresta do tipo *was triggered by* oculta a existência de algum artefato usado pelo processo P_2 gerado pelo processo P_1 . Nesse caso, as arestas inferidas do tipo *used* e *was generated by* são declaradas no mesmo contexto descritivo da aresta do tipo *was triggered by*.

A regra de completude permite estabelecer a existência de algum artefato, mas não determina a identificação dele. Esse fato é consequência de se empregar uma aresta do tipo *was triggered by*, que pode ser considerado um “resumo com perdas” da composição das arestas do tipo *used* e *was generated by*.

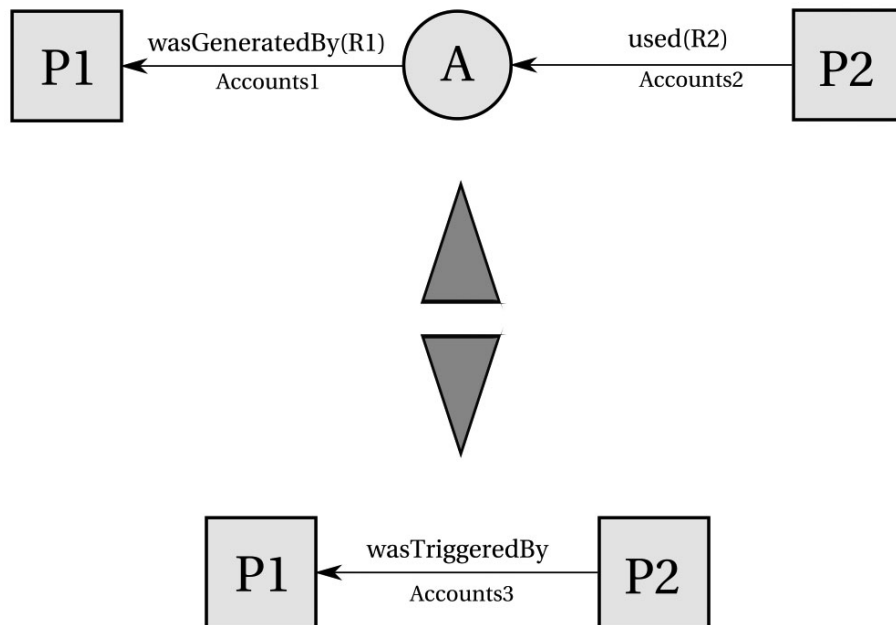


Figura 3.10: Completude: introdução e eliminação de artefatos [54]

Na Figura 3.11 existe apenas uma regra de completude possível, denominada “introdução de processo”. Uma aresta do tipo *was derived from* oculta a existência de um processo intermediário. A completude inversa não pode, porém, ser sustentada, uma vez que sem nenhum conhecimento interno a respeito do processo P é impossível determinar se há uma dependência de dados real entre os artefatos A_1 e A_2 .

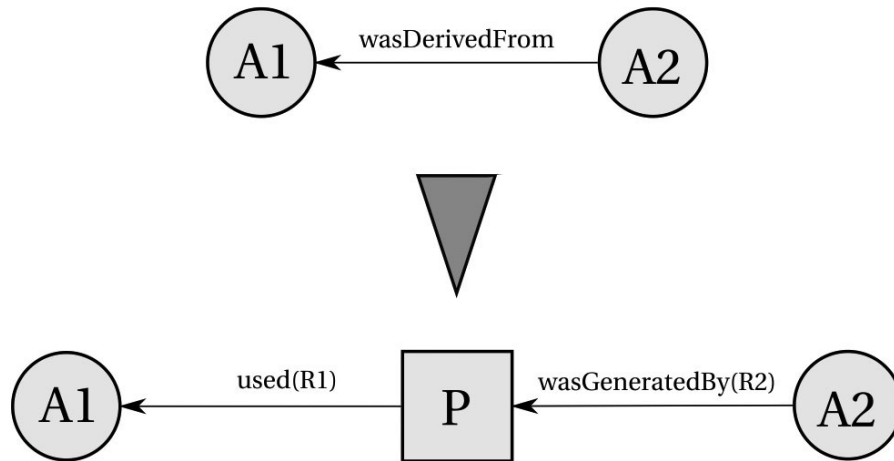


Figura 3.11: Completude: introdução de processo [54]

3.6 Anotações

O OPM permite anexar informações adicionais às entidades do modelo. O objetivo é disponibilizar um recurso capaz de promover uma melhor interoperabilidade ao adicionar significado semântico às informações de proveniência coletadas. Para acomodar tais informações extras de uma maneira extensível, o OPM admite que todas as entidades do modelo possam receber anotações através de um arcabouço específico.

3.6.1 Arcabouço de Anotações

O arcabouço de anotações do Modelo OPM é definido de acordo com as seguintes regras:

1. Uma anotação OPM consiste em uma classe de objetos distinta em relação às demais entidades do modelo.
2. As entidades que podem ser anotadas são grafo, nó, aresta, descrição, papel e a própria anotação.
3. Uma entidade pode estar associada a uma ou mais anotações.
4. Uma instância de uma anotação consiste em um objeto da classe *Anotação OPM* e compõe-se de:
 - Um sujeito, que consiste em uma entidade que pode receber uma anotação e que é identificada de forma única por um identificador.
 - Um conjunto não-vazio de pares propriedade-valor.
 - Uma lista que contém as descrições onde a entidade anotada é referenciada.

O propósito de um *par propriedade-valor* consiste em acrescentar informações adicionais à entidade anotada (o sujeito) no contexto de uma ou mais descrições (*accounts*). O OPM permite a ocorrência de diversos pares propriedade-valor para uma única anotação. É também válida a ocorrência de uma mesma propriedade associada a diferentes valores [54].

A Figura 3.12 mostra como as anotações podem ser incluídas em um grafo de proveniência. O artefato *Bolo* contém uma anotação com a propriedade *Qualidade* e valor *Delicioso* e o artefato *Farinha* contém uma anotação com a propriedade *Tipo* e valor *Com Fermento*.

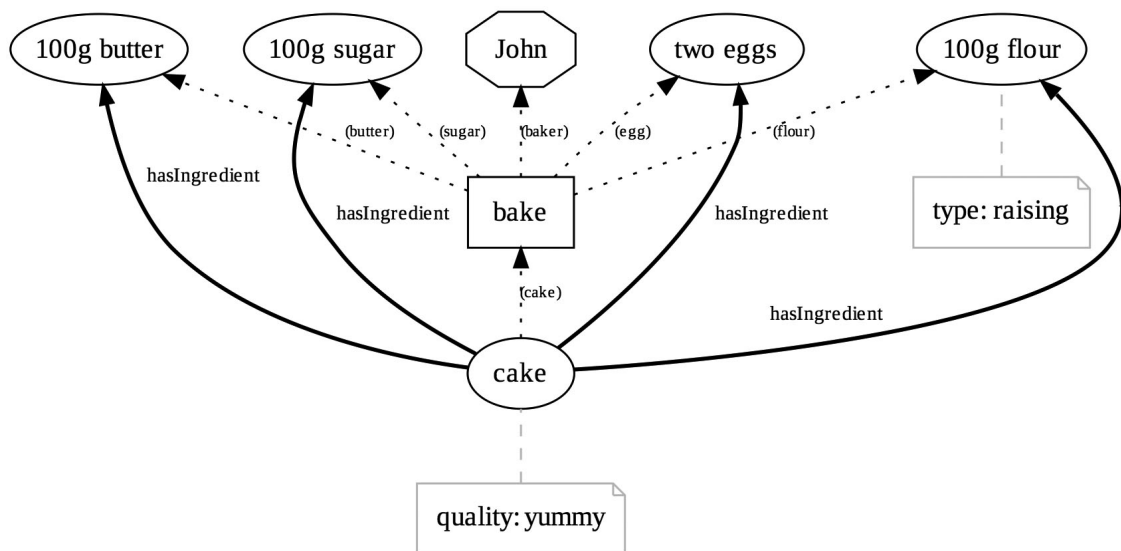


Figura 3.12: Anotações em um grafo de proveniência [54]

3.6.2 Propriedades Fundamentais do OPM

Para fins de interoperabilidade, o OPM define um conjunto de propriedades chamadas fundamentais. É possível identificar cada propriedade através de uma *Uniform Resource Identifier* (URI) que define o tipo esperado para o sujeito e também os valores associados à propriedade. A partir daí, o modelo propõe um significado para cada propriedade fundamental.

Type	<p>Sujeito: Uma entidade anotável</p> <p>Propriedade: <code>http:\\openprovenance.org\\property#type</code></p> <p>Valor: uma URI</p> <p>Descrição: Indica um subtipo para uma entidade OPM. Deve ser representado através de uma URI.</p>
pname	<p>Sujeito: Uma entidade anotável</p> <p>Propriedade: <code>http:\\openprovenance.org\\property#pname</code></p> <p>Valor: uma URI</p> <p>Descrição: Indica um nome que pode ser usado em consultas a grafos de proveniência com o objetivo de comparar entidades OPM. O escopo do nome tenciona-se ser global.</p>
label	<p>Sujeito: Uma entidade anotável</p> <p>Propriedade: <code>http:\\openprovenance.org\\property#label</code></p> <p>Valor: uma string</p> <p>Descrição: Essa propriedade fornece uma versão legível (por seres humanos) de uma entidade OPM.</p>
value	<p>Sujeito: Um artefato</p> <p>Propriedade: <code>http:\\openprovenance.org\\property#value</code></p> <p>Valor: um valor tipado</p> <p>Descrição: Indica uma serialização de um valor de aplicação associado a uma entidade OPM. Tecnologias para serialização incluem XML e n-triplas.</p>
encoding	<p>Sujeito: Um artefato ou um grafo OPM</p> <p>Propriedade: <code>http:\\openprovenance.org\\property#encoding</code></p> <p>Valor: uma URI</p> <p>Descrição: Define como uma serialização foi construída.</p>
profile	<p>Sujeito: Um grafo OPM</p> <p>Propriedade: <code>http:\\openprovenance.org\\property#profile</code></p> <p>Valor: uma URI</p> <p>Descrição: Propriedade aplicável a grafos OPM e que indica o perfil suportado pelo grafo.</p>

3.7 Perfis OPM

O OPM consiste em uma estrutura de representação de alto nível para a proveniência. Nesse contexto, o projeto reconhece que algumas comunidades irão desenvolver as suas *melhores práticas e diretrizes* para uso próprio do modelo. No sentido de encorajar essa noção de boas práticas o modelo propõe a sua formalização a partir do conceito de *perfil OPM*.

Um perfil OPM destina-se a definir uma especialização do modelo e, portanto, deve preservar a compatibilidade com a semântica do OPM. A implicação é que um grafo OPM compatível com um perfil deve preservar as características de um grafo OPM. Isto significa que todas as inferências permanecem válidas em um perfil compatível com um grafo OPM. Portanto, qualquer extensão ao modelo que não preserve a semântica OPM não deve ser definido como um perfil e não deve sequer ser referido como OPM.

Um perfil OPM consiste dos seguintes elementos:

1. Um identificador obrigatório, único e global, que deve ser usado como `value` no perfil `property` em uma anotação de um grafo de proveniência compatível com o perfil.
2. Um vocabulário controlado para anotações (opcional) que consiste em uma especificação das propriedades, sujeitos e valores permitidos.

Esse vocabulário controlado pode ser utilizado com os seguintes propósitos:

- (a) Subtipagem de arestas e nós em grafos de proveniência através da propriedade `type`.
- (b) Definição de propriedades específicas no domínio de uma aplicação.

Por exemplo, uma propriedade referente à posição anexada aos nós pode ser explorada por uma ferramenta de visualização para gerar grafos OPM.

3. Uma orientação geral para expressar gráficos OPM (opcional).

Existem diversas formas de se descrever uma execução através do OPM. Por questões de interoperabilidade, é interessante estabelecer uma diretriz para a estruturação dos grafos de proveniência. Por exemplo, pode ser válido identificar vários tipos de descrições (alto nível, baixo nível, etc.) e determinar que cada descrição contenha arestas de subtipos específicos.

4. Regras para expansão de perfis (opcional).

Em alguns casos, é possível definir-se somente parte dos nós e arestas relacionados a uma execução e usar os recursos de inferência para a derivação dos demais. Portanto, perfis podem conter *regras de expansão* para converter um grafo OPM compatível com um perfil em outro grafo OPM também compatível. Esse mecanismo é denominado *expansão de perfil* e o resultado denomina-se *grafo de perfil expandido*.

5. Sintaxe específica para serialização (opcional). Um perfil pode definir atalhos sintáticos para serializações específicas.

Segundo Moreau *et al.* [54], encontram-se em curso trabalhos de pesquisa com o objetivo de definir diversos perfis úteis (por exemplo, Dublin Core e Coleções). Espera-se que os vocabulários controlados, padrões e regras de inferência possam ser todos expressos em alguma linguagem declarativa. Tal recurso poderia ser utilizado para verificar automaticamente se um gráfico OPM é compatível com um determinado perfil.

4 SCIENTIFIC WORKFLOW PROVENANCE SYSTEM

O objetivo do presente trabalho consiste em apresentar uma arquitetura para um sistema de proveniência de dados e processos no contexto de experimentos científicos processados através de simulações computacionais em ambientes de pesquisa colaborativa, dispersos geograficamente e interconectados através de uma grade computacional.

Especificamente, propõe-se um modelo para um sistema de proveniência de dados e processos cujo propósito consiste em interagir com os sistemas de gerenciamento de *workflows* científicos utilizados em um ambiente colaborativo com a finalidade de capturar e gerir as informações de linhagem geradas em um contexto de *e-Science*.

A arquitetura apresentada denomina-se *Scientific Workflow Provenance System (SciProv)*. Esta aceção justifica-se por destacar algumas das palavras-chave relacionadas ao tema abordado no trabalho.

4.1 Arquitetura do *SciProv*

A concepção da arquitetura apresentada baseia-se em alguns requisitos considerados fundamentais ao delimitar-se o escopo do trabalho:

1. Independência em relação aos modelos de dados e processos adotados pelos SGWfC existentes.
2. Aplicabilidade direcionada — mas não restrita — a experimentos científicos, incluindo *workflows* executados em ambientes heterogêneos e dispersos em grades computacionais.
3. Emprego de tecnologias da *Web Semântica* — RDF, OWL, ontologias, máquinas de inferência — para representação e consulta aos dados de proveniência.
4. Possibilidade de ajuste no mecanismo de captura para permitir o controle sobre o impacto dos processos de coleta e persistência dos dados de proveniência no desempenho da execução dos experimentos científicos.

Um requisito adicional que a arquitetura do *SciProv* procura investigar consiste na adequabilidade da interação do sistema com o usuário, em geral um pesquisador de áreas diversas da Ciência. Nesse cenário, procura-se considerar o perfil do usuário em aspectos relacionados à proficiência no uso de ferramentas computacionais complexas como é o caso da linguagem de consulta SPARQL.

4.1.1 Componentes da Arquitetura

Uma representação da arquitetura do *SciProv* considerando-se um cenário típico de aplicação é apresentado na Figura 4.1. Em um ambiente colaborativo interconectado através de uma grade computacional, os experimentos podem ser conduzidos de forma conjunta por grupos de cientistas que se encontram em centros de pesquisa localizados remotamente.

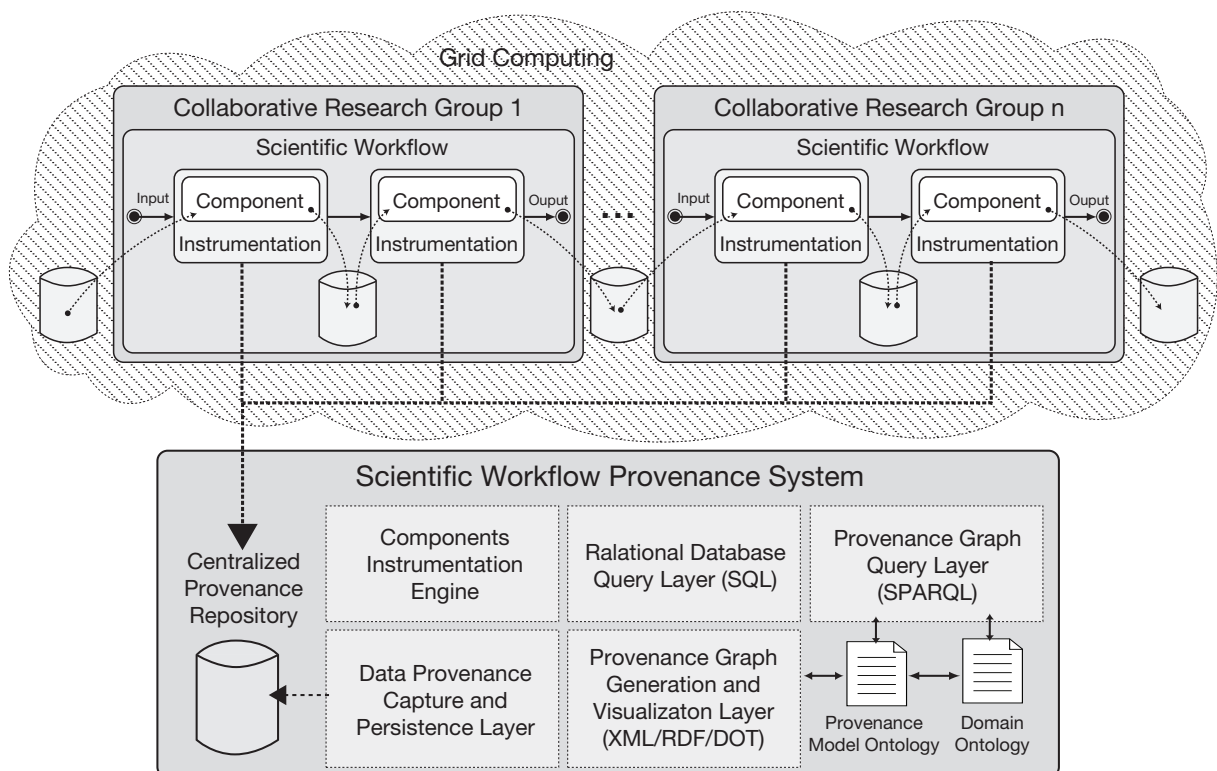


Figura 4.1: Arquitetura do *SciProv*

Os pesquisadores podem modelar *workflows* científicos a partir de SGWfCs distintos (como Kepler, Taverna e Vistrails) e cuja execução requer o acesso a repositórios heterogêneos (dados relacionais, semiestruturados, etc.) e distribuídos em uma grade computacional. Informações armazenadas originalmente em repositórios heterogêneos e *workflows* científicos modelados em SGWfCs diferentes carecem de mecanismos que contribuam

para uma maior interoperabilidade entre os dados gerados e os processos orquestrados no âmbito de projetos de pesquisa colaborativos.

A responsabilidade inicial do *SciProv* consiste em prover um mecanismo de instrumentalização para diversos componentes dos *workflows* científicos envolvidos na condução do experimento colaborativo cujos dados de proveniência necessitam de coleta para posterior análise.

De forma concreta, um mecanismo de instrumentalização é implementado a partir de tecnologia de serviços *web* e configurado manualmente para cada componente cuja proveniência deve ser coletada. Esse mecanismo tem como objetivo capturar informações geradas durante o processo de execução do *workflow* e enviar esses metadados para um repositório centralizado de proveniência gerenciado pelo *SciProv*.

As informações de proveniência coletadas são persistidas em um sistema de gerenciamento de base de dados relacional (SGBDR). O *SciProv* contém uma “camada de consulta à base de dados relacional” (*Relational Database Query Layer* na Figura 4.1) que provê uma interface com o usuário para a entrada de consultas formuladas na linguagem padrão para o modelo relacional.

A “camada de geração e visualização do grafo de proveniência” (*Provenance Graph Generation and Visualization Layer* na Figura 4.1) possui diversas atribuições inter-relacionadas. A partir da base de dados relacional, as informações persistidas segundo o modelo OPM são processadas com o objetivo de obter-se uma representação em memória do grafo de proveniência correspondente a cada execução de um *workflow* científico. Esta camada permite ainda a construção de uma representação visual do grafo de proveniência gerado. Este processo inicia-se a partir da serialização do grafo de proveniência na sintaxe de uma linguagem de marcação. Este arquivo constitui a base para a conversão do grafo para um linguagem específica de descrição de gráficos, a partir da qual é possível produzir saídas em um formato padrão de visualização.

Por último, a “camada de consulta ao grafo de proveniência” (*Provenance Graph Query Layer* na Figura 4.1) tem por objetivo prover um mecanismo e uma interface para que o usuário formule consultas a partir de uma linguagem de consulta da *web* semântica. Essa camada está associada às ontologias do modelo OPM de proveniência e do domínio estudado. Esse recurso confere ao *SciProv* a possibilidade de processar consultas a partir de máquinas de inferência, capazes de efetuar deduções sobre essas

bases de conhecimento e obter resultados importantes ao extrair informações adicionais além daquelas que encontram-se registradas de forma explícita nos grafos de proveniência gerados.

Estas duas últimas camadas apresentadas — *Provenance Graph Generation and Visualization Layer* e *Provenance Graph Query Layer* — englobam os maiores diferenciais do trabalho desenvolvido. Em um contexto de *e-Science*, a ênfase na concepção e construção da arquitetura do *SciProv* consiste em empregar recursos da *web* semântica para oferecer ao pesquisador um ambiente fundamentado na interoperabilidade para a gerência e a consulta à proveniência de dados heterogêneos e distribuídos.

As subseções seguintes descrevem os componentes que compõem a arquitetura do *SciProv* conforme apresentada na Figura 4.1.

4.1.2 Mecanismo de Instrumentalização

O *SciProv* apresenta uma abordagem para o gerenciamento da proveniência de dados de forma independente de SGWfCs. Isso implica dizer que o *SciProv* é responsável pela coleta dos dados de proveniência gerados. Para que essa abordagem seja possível, torna-se necessário especificar um mecanismo de *instrumentalização* cujo objetivo consiste em coletar os dados de linhagem durante o processo de execução de um *workflow* científico modelado a partir de um SGWfC.

A arquitetura emprega um mecanismo de instrumentalização baseado em tecnologia de serviços *web*. A escolha de uma solução a partir desse modelo de tecnologia tem como pressupostos os seguintes fatores:

- Suporte ao desenvolvimento de artefatos de *software* para execução em ambiente distribuído e heterogêneo interconectado a partir de uma grade computacional.
- Alicerce em protocolos abertos e consensuais para a internet.
- Interoperabilidade entre os artefatos construídos utilizando-se serviços *web*.

Especificamente, o *SciProv* adota uma solução para a construção de serviços *web* a partir do protocolo *Simple Object Access Protocol* (SOAP), que se baseia na invocação remota de um método a partir da especificação do endereço do componente, além do nome do método e dos respectivos argumentos.

O modelo SOAP não estabelece ou impõe qualquer semântica, seja sobre o modelo de programação, a plataforma de *hardware* ou o protocolo de transporte. Esta característica

é relevante no contexto do presente trabalho pois permite que tanto o serviço *web* quanto o cliente que invoca o serviço sejam artefatos de *software* que podem ser desenvolvidos utilizando-se diferentes linguagens de programação. Isso permite também uma melhor interoperabilidade e intercomunicação em um ambiente heterogêneo e distribuído.

É adequado observar que algumas características do protocolo SOAP — independência de protocolo de transporte e concepção do modelo orientado a sistemas heterogêneos e distribuídos — tornam-o mais interessante no cenário proposto para o presente trabalho em comparação com a tecnologia RESTful, baseada na arquitetura *Representational State Transfer* (REST).

Uma particularidade importante do modelo de instrumentalização do *SciProv* refere-se a adoção de um único serviço *web* para processar o encapsulamento de quaisquer componentes originais de um experimento modelado a partir de um *workflow* científico, conforme apresentado na Figura 4.2.

Pode-se observar que o serviço *web* do *SciProv* pode ser invocado uma ou mais vezes para cada componente original do *workflow*. Em um cenário típico, uma primeira instância do serviço *web* é configurada para coletar os dados de entrada (*artefatos*, segundo o modelo OPM) a serem processados no componente original. Em uma segunda instância, são capturadas as informações referentes ao componente original (um *processo*, segundo o modelo OPM) tais como início e fim da execução, dependências causais relacionadas aos artefatos de entrada e de saída (necessários para a caracterização explícita de que o componente *usou* os dados de entrada e que *gerou* os dados de saída). Na Figura 4.2 uma terceira instância captura os dados (*artefatos*) gerados na saída do componente executado.

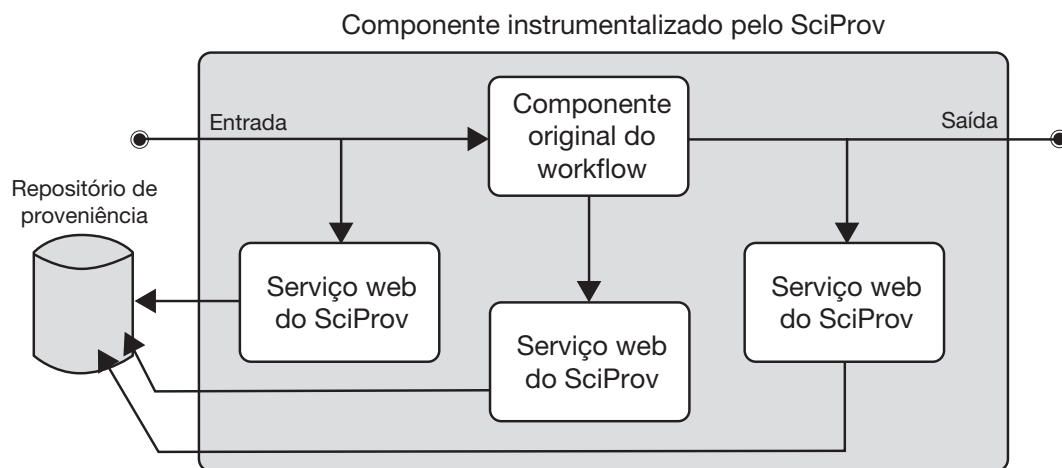


Figura 4.2: Mecanismo de instrumentalização do *SciProv*

É importante observar que pode ser utilizado um número ainda maior de instâncias do serviço *web* de instrumentalização para um mesmo componente original, dependendo da necessidade de coletar outras entidades e dependências causais do modelo OPM identificadas em um *workflow*.

Por hipótese, a instrumentalização na forma adotada pelo presente trabalho é possível em qualquer sistema de gerenciamento de *workflows* científicos que ofereça suporte a serviços *web* no padrão SOAP.

4.1.3 Componente Composto

No contexto do *SciProv*, o processo de instrumentalização de componentes deve ser realizado manualmente no SGWfC utilizado para a orquestração do *workflow* científico. Concretamente, é necessário que um assistente ou colaborador familiarizado com o uso do SGWfC em questão implemente um componente composto conforme apresentado na Figura 4.2 capaz de encapsular as chamadas ao serviço *web* de instrumentalização do *SciProv*.

Em um ambiente de pesquisa colaborativo pode-se considerar que os experimentos modelados computacionalmente poderão ser acessados por pessoal qualificado e com conhecimento do processo de instrumentalização no sentido de implementar o mecanismo necessário para a captura dos dados de proveniência.

Como resultado do processo de instrumentalização, o componente original é substituído por um *componente composto* na visão do usuário do *workflow* científico, conforme apresentado na Figura 4.3. Esse modelo permite abstrair do cientista toda a complexidade relacionada ao processo de instrumentalização incorporado ao experimento científico.

Para que o processo de instrumentalização empregado na arquitetura do *SciProv* possa utilizar o conceito de componente composto é necessário que o SGWfC utilizado suporte tal recurso. Nesse aspecto, alguns SGWfCs como Kepler, Taverna e Vistrails provêm funcionalidades que implementam o conceito de componente composto.

4.1.4 Instrumentalização em Diversos Níveis de Detalhamento

A partir de uma abordagem de coleta de proveniência em nível de processo (componente), abre-se a possibilidade de controle sobre a granularidade dos dados a serem capturados. No campo da experimentação científica, pode ser importante para o pesquisador analisar

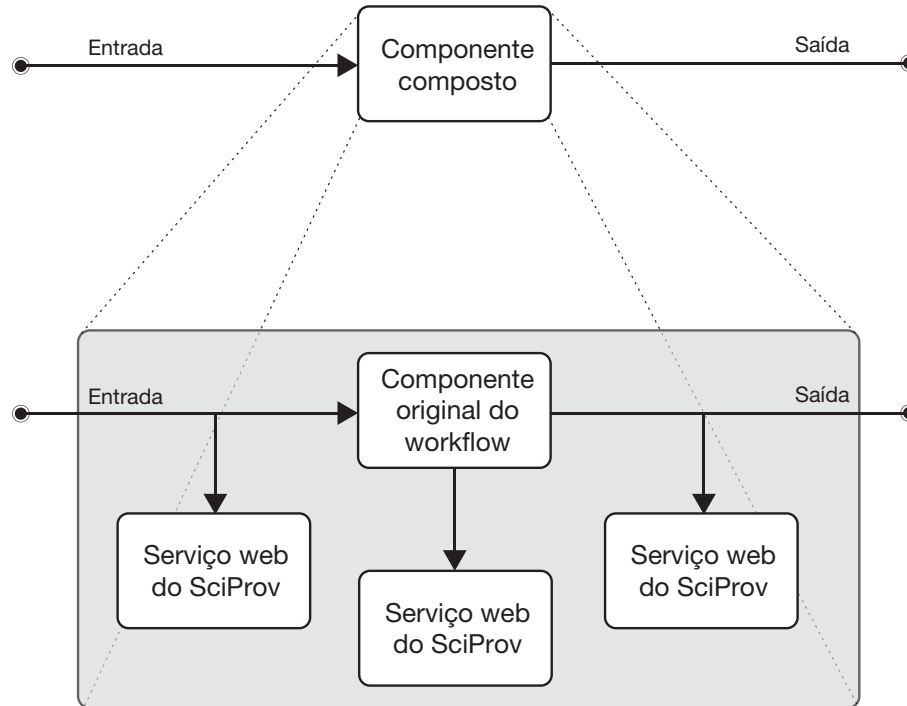


Figura 4.3: Abstração do Mecanismo de Instrumentalização

informações e elaborar consultas aos dados de linhagem em diversos níveis de detalhamento.

Nesse cenário, torna-se relevante que o modelo de proveniência possa ser capaz de tratar as informações de proveniência em níveis distintos de granularidade. O OPM provê este recurso de forma conveniente a partir da entidade denominada “descrição” (*account*). Considerando-se a execução de um *workflow* científico modelado a partir de um SGWfC, o OPM permite o tratamento das informações de proveniência em diversos níveis de detalhamento, onde cada nível é associado a uma descrição distinta.

O suporte oferecido pelo modelo OPM à proveniência em descrições distintas é integralmente adotado pela arquitetura do *SciProv*. Pode-se afirmar que não há restrição teórica quanto ao número de níveis de detalhamento para a proveniência de dados que podem ser capturados pelo *SciProv* a partir da execução de um experimento científico modelado em um SGWfC com suporte a componentes compostos.

De fato, a possibilidade de captura da proveniência em diversos níveis de granularidade provida pelo modelo OPM e suportada pelo *SciProv* adequa-se de forma consistente com o suporte provido pelos SGWfC a um modelo de abstração em camadas conforme apresentado na Figura 4.4.

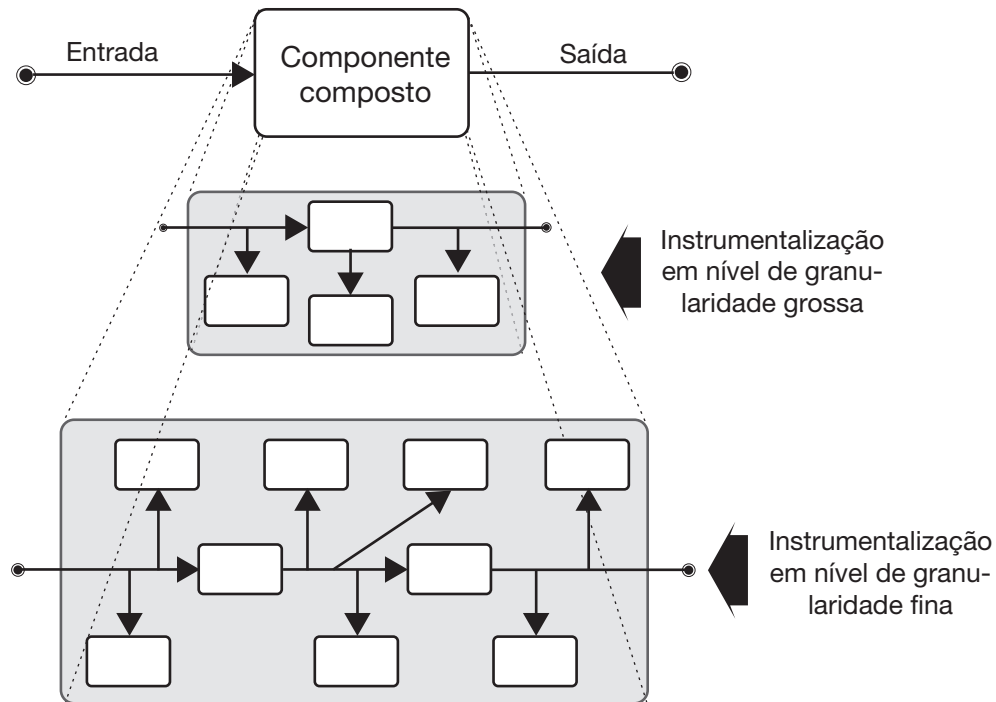


Figura 4.4: Exemplo de componente composto em dois níveis de abstração

4.1.5 Importância do Mecanismo de Coleta em Camadas

A partir do mecanismo de instrumentalização em diversas descrições OPM e de forma coerente com o modelo de abstração em camadas para a coleta da proveniência, é possível configurar um experimento para capturar as informações de linhagem em níveis de granularidade ou especificidade distintos. Nesse contexto, cada usuário pode utilizar os metadados no nível de detalhe ou contexto mais adequado à pesquisa científica com a qual esteja envolvido.

Também pode-se observar que descrições distintas de proveniência referentes a uma mesma execução de um *workflow* científico representam semânticas diversas de linhagem. Portanto, a partir da coleta de dados em níveis diversos de granularidade é possível proceder análises e consultas de proveniência distintas, de acordo com as particularidades das pesquisas em andamento.

Nesse cenário, deve-se destacar que a estratégia de coleta e gerência da proveniência em diversos níveis de abstração pode ser considerada como um importante diferencial da arquitetura do *SciProv* por constituir-se em um mecanismo com diversos desdobramentos e potencialidades:

- Permite parametrizar a consulta à proveniência de acordo com o foco específico de

uma pesquisa. Nesse contexto, pode-se utilizar apenas um conjunto de metadados representado em uma descrição OPM.

- Permite processar consultas de proveniência relacionadas a uma mesma execução de um *workflow* com a possibilidade de resultados com maior ou menor abrangência e detalhamento.
- Constitui um mecanismo para controle do desempenho de processamento de um *workflow* na medida em que apenas os níveis de abstração necessários em um determinado escopo de trabalho sejam habilitados para a coleta de proveniência.
- De forma análoga, constitui um mecanismo de controle sobre o volume de metadados de proveniência coletados e armazenados. Em um ambiente de pesquisa colaborativa onde *workflows* complexos têm potencial para processar grande quantidade de dados, é importante dispor de alternativas para adequar a captura e gerência da proveniência aos recursos disponíveis para esse fim.
- Facilita o processo de instrumentalização, no sentido de permitir a modularização do trabalho. Esse enfoque proporciona vantagens em termos de criação e manutenção dos componentes compostos de adaptação do *workflow*.

Dentre os trabalhos relacionados não se observa mecanismos de coleta da proveniência capazes de proporcionar um conjunto de funcionalidades e potencialidades conforme representadas na arquitetura do *SciProv* com base no modelo de abstração em camadas.

4.1.6 Persistência dos Dados de Proveniência

Para a persistência dos metadados de proveniência, a maioria dos SGWfCs têm adotado soluções a partir do modelo relacional, embora existam alguns modelos que utilizem documentos semiestruturados baseados em linguagem XML [16]. Barga e Digiampietri [38] defendem o modelo relacional como a alternativa mais adequada para o armazenamento dos metadados de proveniência no contexto de *e-Science*. A argumentação baseia-se nos quesitos de confiabilidade e independência em relação a SGWfCs.

A arquitetura do *SciProv* adota uma solução a partir de uma base de dados relacional para prover a persistência dos dados de proveniência coletados. Embora o *SciProv* adote um SGBDR específico, a escolha da infraestrutura computacional de suporte ao modelo relacional pode ser alterada sem implicações ou restrições no âmbito da arquitetura do

SciProv. Em outros termos, o esquema da base de dados conforme modelado na arquitetura pode ser configurado para outras soluções disponíveis para o armazenamento e a gerência de dados no modelo relacional.

O modelo conceitual que representa a descrição do banco de dados relacional definido na arquitetura do *SciProv* é apresentado na Figura 4.5 na forma de um diagrama entidade-relacionamento (DER).

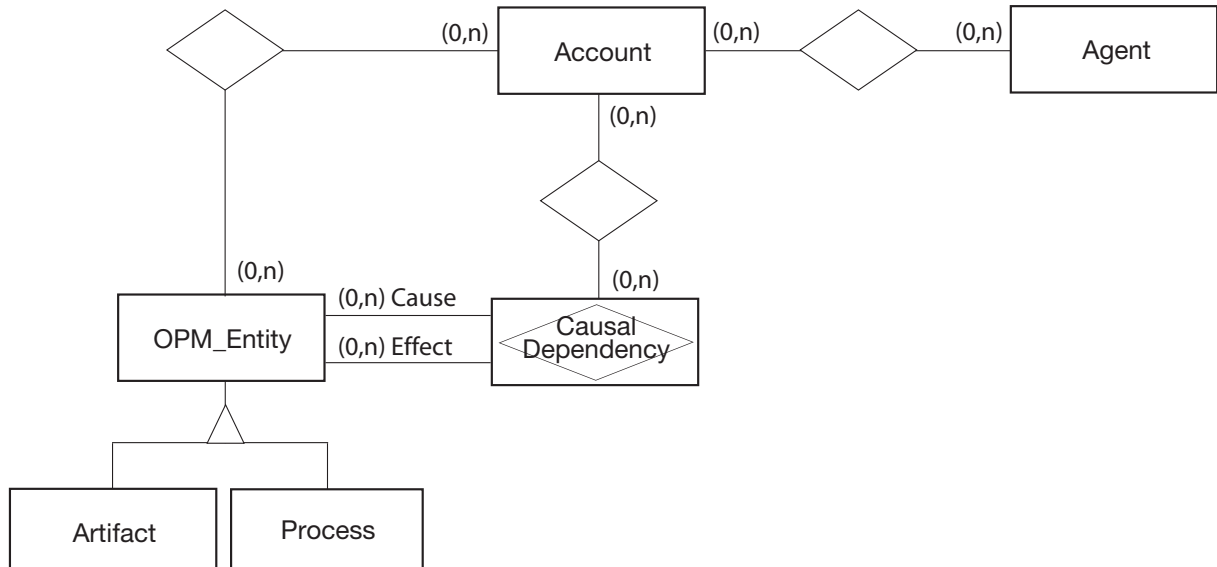


Figura 4.5: *SciProv* – DER que representa o modelo conceitual

Pode-se verificar que o DER representado na Figura 4.5 identifica as entidades do modelo conceitual (entidades e relações de dependência causal segundo o modelo de proveniência OPM). Pode-se observar que tanto a entidade *Artifact* quanto a entidade *Process* possuem dois tipos distintos de relacionamento com a entidade *Causal Dependency*, denominados *Cause* e *Effect*.

A partir do modelo conceitual, a arquitetura do *SciProv* define um modelo lógico para a base de dados composto por cinco tabelas, conforme apresentado na Figura 4.6.

A tabela ACCOUNT tem por objetivo armazenar os dados referentes à entidade OPM denominada “Descrição” (*account*) para cada execução de um determinado *workflow* científico instrumentalizado e cujos dados de proveniência devem ser coletados no contexto do *SciProv*.

A tabela AGENT deve armazenar os dados referentes à entidade “Agente” para cada execução de um *workflow* científico. Além disso, é possível incluir uma anotação para cada entidade persistida do tipo agente a partir do campo LABEL_VALUE.

```

ACCOUNT(ACCOUNT_ID, WORKFLOW_REF, ACCOUNT_VALUE)
AGENT(AGENT_ID, WORKFLOW_REF, AGENT_VALUE)
ACCOUNT_AGENT(ACCOUNT_ID, AGENT_ID, LABEL_VALUE)
CAUSAL_DEPENDENCY(CAUSAL_DEPENDENCY_ID, WORKFLOW_REF,
    ROLE_VALUE, TIME_NOLATERTHAN, TIME_NOEARLIERTHAN)
ACCOUNT_CAUSAL_DEPENDENCY(CAUSAL_DEPENDENCY_ID, ACCOUNT_ID)
CAUSE(OPM_ENTITY_ID, CAUSAL_DEPENDENCY_ID, ACCOUNT_ID, CAUSE_VALUE)
EFFECT(OPM_ENTITY_ID, CAUSAL_DEPENDENCY_ID, ACCOUNT_ID, EFFECT_VALUE)
OPM_ENTITY(OPM_ENTITY_ID, WORKFLOW_REF)
ARTIFACT(OPM_ENTITY_ID, LABEL_VALUE)
PROCESS(OPM_ENTITY_ID, LABEL_VALUE)

```

Figura 4.6: *SciProv* – Tabelas do modelo lógico

A tabela `ACCOUNT_AGENT` tem por objetivo armazenar os relacionamentos entre as entidades OPM “Descrição” e “Agente”.

A tabela `CAUSAL_DEPENDENCY` tem por objetivo prover a persistência das relações de dependências causais identificadas no *workflow* instrumentalizado, de acordo com o modelo OPM. Cada dependência causal representa uma aresta no grafo de proveniência e, portanto, liga duas entidades do tipo “Agente”, “Processo” ou “Artefato”. Os campos `TIME_NOLATERTHAN` e `TIME_NOEARLIERTHAN`, opcionais, estão relacionados às restrições temporais e de observação do tempo conforme especificado no modelo OPM (Seção 3.4). Assim, de acordo com o tipo de dependência causal coletado, apenas um ou ambos os campos são utilizados. O campo `ROLE_VALUE`, opcional, refere-se ao “Papel” (*role*) associado à dependência causal persistida. Um papel tem por objetivo fazer a distinção entre diversas relações de dependência causal associadas a uma mesma entidade do tipo Processo, conforme apresentado na Seção 3.2.3.

A tabela `ACCOUNT_CAUSAL_DEPENDENCY` deve armazenar os relacionamentos entre a entidade “Descrição” e as relações de dependências causais observadas no grafo OPM. A Tabela `CAUSE` armazena os relacionamentos do tipo “Causa” observados entre entidades (Agente, Processo ou Artefato) e relações de dependência causal (*used*, *wasGeneratedBy*, *wasControlledBy*, *wasTriggeredBy* e *wasDerivedBy*). A Tabela `EFFECT` armazena os relacionamentos do tipo “Efeito” observados entre entidades OPM (Agente, Processo ou Artefato) e relações de dependência causal.

A tabela `ARTIFACT` armazena os dados referentes à entidade “Artefato” para cada execução de um *workflow*. De forma análoga, a tabela `PROCESS` armazena os dados referentes à entidade “Processo” para cada execução de um *workflow*. Uma anotação

pode ser incluída a partir do campo *Label_Value* para ambas as tabelas.

Além das tabelas apresentadas anteriormente e que representam o modelo lógico definido pela arquitetura do *SciProv* para a persistência dos metadados de proveniência, há ainda uma tabela adicional referente aos dados de anotações (Figura 4.7). Conforme discutido na Seção 3.6, o modelo OPM admite a inclusão de anotações às entidades e dependências causais em um grafo de linhagem. O recurso é implementado através de um arcabouço próprio, onde as anotações são tratadas como entidades independentes e específicas para esse fim [54].

ANNOTATION(ANNOTATION_ID, WORKFLOW_REF, ACCOUNT_REF,
OPM_REF, ANNOTATION_TYPE, ANNOTATION_VALUE)

Figura 4.7: *SciProv* – Tabela do arcabouço de anotações

Cada tupla da tabela ANNOTATION exibida na Figura 4.7 refere-se a uma anotação declarada no *workflow* científico modelado. O campo ANNOTATION_TYPE registra o tipo de anotação no arcabouço e o campo ANNOTATION_VALUE relaciona-se ao valor assumido para o tipo de anotação informado.

Um outro aspecto relevante no contexto do presente trabalho refere-se ao modelo de armazenamento. A proposta original para a arquitetura do *SciProv* baseia-se em um modelo de armazenamento distribuído, onde os dados de proveniência sejam persistidos no nó da grade computacional onde forem coletados.

Uma abordagem baseada em um modelo distribuído de armazenamento pode ser interessante no contexto proposto para o presente trabalho, relacionado à experimentação científica em um ambiente colaborativo e disperso geograficamente. A hipótese é que a alocação de repositórios de proveniência diretamente nos nós da grade computacional onde se encontram os dados — no todo ou em parte — a serem processados pode contribuir para a melhoria do tempo de execução dos *workflows* instrumentalizados.

Entretanto, o enfoque distribuído para o modelo de armazenamento implica em uma série de desafios para a arquitetura do *SciProv* [55]:

- *Complexidade* — trabalho extra pode ser requerido para garantir a transparência da natureza distribuída dos dados.
- *Custo de implantação* — mesmo em um ambiente colaborativo, o aumento da complexidade e uma infraestrutura mais extensa podem implicar em maiores custos para implantação e manutenção do modelo distribuído.

- *Segurança* — banco de dados distribuídos também devem ser seguros e, em uma grade computacional, os requisitos de segurança tornam-se mais críticos.
- *Integridade* — manter a integridade dos dados em um modelo de persistência distribuído em uma grade computacional pode torna-se uma meta de difícil implementação.

Em razão das implicações apresentadas para a adoção de um modelo de armazenamento distribuído, a arquitetura do *SciProv* provê inicialmente o suporte a um modelo centralizado para a persistência dos metadados de proveniência coletados. Nessa abordagem simplificadora, utiliza-se um único nó coordenador para o armazenamento dos dados de proveniência coletados.

Essa solução apresenta benefícios em termos de simplicidade no que concerne à gerência e à consulta aos dados de proveniência persistidos. Entretanto, em cenários de grande volume de dados manipulados durante a execução do experimento científico, o tráfego de informações na grade computacional referentes à coleta de linhagem pode impactar o desempenho do *workflow* científico.

Nesse contexto, sugere-se que o *SciProv* possa avaliar em etapas posteriores um modelo de persistência para os dados de proveniência de forma distribuída, no qual os metadados sejam armazenados no próprio nó da grade computacional onde foram originalmente capturados. Espera-se que trabalhos futuros desenvolvidos no âmbito do Mestrado em Modelagem Computacional da UFJF possam aprofundar os estudos referentes a este aspecto do modelo e refinar a arquitetura desenvolvida.

Uma vez que as informações de proveniência coletadas são armazenadas em uma base de dados relacional, a arquitetura do *SciProv* provê uma interface com o usuário para a consulta aos dados de linhagem a partir da linguagem padrão *Structured Query Language* (SQL).

É conveniente considerar, entretanto, que o suporte à consulta em SQL aos metadados de proveniência não constitui uma característica relevante no contexto da arquitetura do *SciProv*. Trata-se de um mecanismo adicional, cuja argumentação para inclusão no projeto baseia-se nos seguintes aspectos:

- Constitui-se na linguagem padrão para consulta a bases de dados relacionais, conforme é o caso dos repositórios do *SciProv*.

- Permite a formulação de consultas aos metadados de proveniência, embora limitadas às restrições impostas pelo modelo relacional. Em outros termos, consultas em SQL não podem tirar proveito da infraestrutura baseada em *web* semântica presente na arquitetura do *SciProv*.

Nesse contexto, é importante ressaltar que consultas em linguagem SQL aos metadados de proveniência persistidos em um modelo relacional permitem retornar apenas as informações explicitamente declaradas a partir do mecanismo de instrumentalização inserido em um *workflow* científico.

A arquitetura do *SciProv* inclui uma camada (ver seção 4.2.10) que emprega recursos da *web* semântica para a consulta aos dados de proveniência. O uso desta tecnologia permite processar inferências sobre a linhagem dos dados e obter resultados importantes ao extrair informações adicionais além daquelas que encontram-se registradas de forma explícita na base de dados relacional.

4.1.7 Geração do Grafo de Proveniência

Uma das características mais relevantes da arquitetura do *SciProv* consiste em prover uma infraestrutura computacional baseada em tecnologia *web* semântica para processar os dados de proveniência. O propósito dessa abordagem é extrair informações adicionais além daquelas que se encontram explicitamente registradas na base de dados relacional.

Nesse sentido, *SciProv* baseia-se na visão proposta pelo projeto *Open Provenance Model*, que consiste em permitir que a proveniência de dados capturada a partir de sistemas heterogêneos possa ser expressa, reunida de forma coerente e consultada de uma maneira integrada [27]. Assim, a visão do OPM provê um conjunto de guias de estilo e melhores práticas para o apoio à interoperabilidade no contexto da proveniência de dados. Esta visão engloba um vocabulário controlado e um conjunto de formatos e APIs para a serialização dos metadados.

A visão proposta pelo projeto *Open Provenance Model* pode ser representada a partir de uma arquitetura em camadas capaz de suportar funcionalidades adicionais além da especificação de um modelo abstrato para a representação da proveniência de dados, conforme apresentado na Figura 4.8 [27].

A camada inferior caracteriza o núcleo do modelo (*OPM Core*), responsável pela definição do modelo abstrato de proveniência. Associado ao núcleo, encontra-se uma subcamada

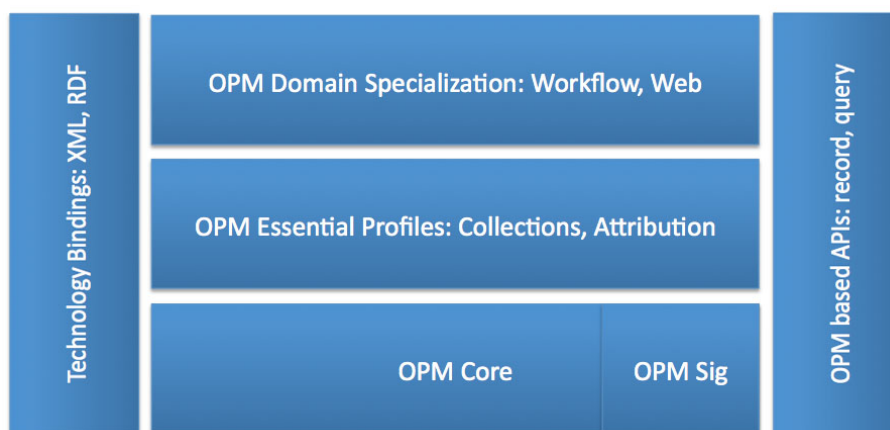


Figura 4.8: Arquitetura em camadas do modelo OPM

de assinatura digital (*OPM Sig*), relacionada à questão da responsabilidade. O intento é prover o endosso de uma determinada informação ou resultado obtido. A camada seguinte, denominada *OPM Essential Profiles* é necessária no sentido de facilitar a adoção do modelo, ao fornecer um conjunto de guias de estilo e perfis genéricos. A camada superior relaciona-se a possíveis especializações do modelo OPM (*OPM Domain Specialization*). Em particular, dois domínios relevantes encontram-se contemplados: *workflows* e a *web*. Em anexo à estrutura em camadas, o modelo identifica a necessidade de tecnologias para a vinculação (*Technology Bindings*) do modelo — XML e RDF já existentes, JSON em planejamento. Há ainda questões em aberto, segundo a visão do projeto *Open Provenance Model* (*OPM based APIs*). Os principais pontos a serem tratados nesse aspecto estão associados à persistência dos metadados e às técnicas para consulta à proveniência.

Vale ainda mencionar que a arquitetura em camadas do projeto OPM é gerenciada a partir de um processo de governança leve e estruturado, inspirado na comunidade de código livre, adotado para acompanhar as atualizações da especificação do modelo. Nesse sentido, o processo de governança visa a tomada de decisões por consenso [56].

O objetivo da camada de geração do grafo de proveniência do *SciProv* consiste em processar as informações de um *workflow* científico persistidas na base de dados relacional e produzir uma representação em memória aderente ao modelo de proveniência OPM.

Conforme apresentado na Seção 3.2.2, o OPM visa capturar as dependências causais entre as entidades do modelo e representar a proveniência na forma de um grafo dirigido cujos nós correspondem a artefatos, processos e agentes e cujas arestas correspondem a dependências causais.

Nesse contexto, a camada *OPM Core* do projeto *Open Provenance Model* (Figura 4.8) fornece mecanismos para permitir um processo de representação padrão do grafo de proveniência em memória. Deve-se observar que a infraestrutura disponibilizada pelo projeto visa prover um conjunto de métodos e técnicas uniformes, cuja intenção é fomentar o emprego normalizado do modelo e contribuir para a interoperabilidade entre as aplicações baseadas no OPM.

4.1.8 Processo de Vinculação em Tecnologia Web Semântica

A partir da modelagem do grafo de proveniência na memória de acordo com as especificações do OPM, a arquitetura do *SciProv* provê recursos para a vinculação dos dados de linhagem em formatos compatíveis com a tecnologia *web* semântica. Nesse contexto, o projeto *Open Provenance Model* (Figura 4.8) provê uma infraestrutura que padroniza e simplifica o processo de serialização do grafo de proveniência a partir da camada *Technology Bindings: XML e RDF*.

Com base no processo de vinculação do grafo de proveniência segundo o OPM, um dos recursos oferecidos pelo *SciProv* consiste em prover um mecanismo para a construção de uma representação gráfica que permita a visualização da linhagem dos dados em um diagrama que apresente as entidades e dependências causais do modelo abstrato de proveniência.

Embora uma representação gráfica do grafo de proveniência não seja relevante no contexto da arquitetura, o recurso oferecido é interessante no sentido de prover ao cientista uma visão intuitiva da linhagem de um determinado dado produzido no contexto de um experimento científico. Uma representação gráfica facilita a contextualização das diversas entidades (artefatos, processos e agentes OPM) e relações de dependências causais explicitados no grafo de proveniência.

Para tanto, o *SciProv* provê os recursos para a conversão do grafo de proveniência serializado em uma linguagem de marcação para a especificação DOT. Este formato consiste em uma linguagem baseada em texto para a descrição de gráficos que utiliza uma notação legível por seres humanos e que permite a interpretação computacional [57].

Dentre diversos outros formatos de gráficos, a linguagem DOT permite descrever grafos acíclicos e dirigidos. A palavra-chave **digraph** é usada para começar um novo grafo. Os nós são descritos entre chaves. Uma seta à direita é usado para mostrar as relações entre os

nós existentes no grafo. A Figura 4.9 apresenta um grafo simples declarado na linguagem DOT.

```
digraph grafo {
    a -> b -> c;
    b -> d;
}
```

Figura 4.9: Exemplo de grafo simples expresso na linguagem DOT

A arquitetura do *SciProv* emprega o grafo de proveniência convertido em sintaxe DOT como formato de entrada para a elaboração de uma representação gráfica em um modelo de visualização padrão, que pode ser associado tanto a aplicações *desktop* quanto *web*. É importante observar que no contexto do *SciProv*, todo o processo de vinculação do grafo de proveniência é aderente às diretrizes formuladas no projeto *Open Provenance Model*, conforme apresentado em [54].

A Figura 4.10 apresenta a representação gráfica em formato padrão *Portable Document File* (PDF) construída a partir da notação em linguagem DOT exibida na Figura 4.9.

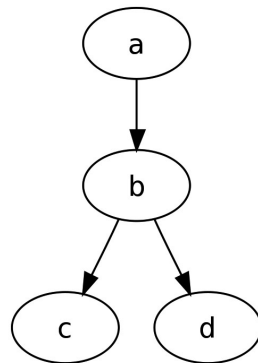


Figura 4.10: Representação gráfica gerada a partir da linguagem DOT

Portanto, pode-se sintetizar o enfoque adotado na arquitetura do *SciProv* para a representação dos dados de proveniência a partir de um modelo em camadas conforme apresentado na Figura 4.11 (ver Seção 4.1.9).

4.1.9 Vinculação em Formato RDF

A representação em formato RDF dos dados de proveniência modelados em memória pelo *SciProv* é essencial para a construção de uma arquitetura baseada na tecnologia *web* semântica.

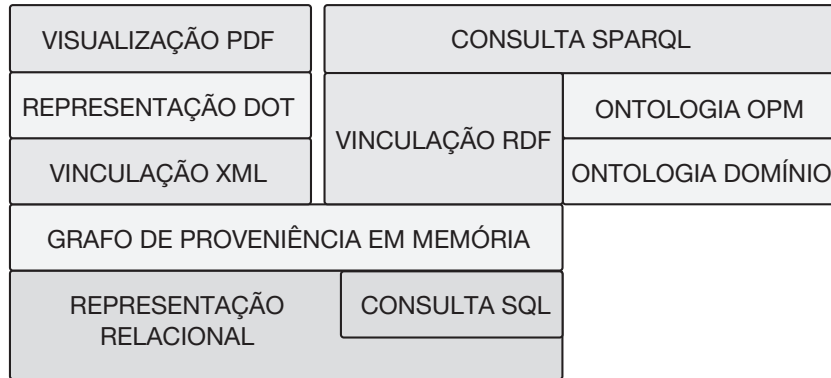


Figura 4.11: Modelo em camadas do *SciProv* para a representação da proveniência

Conforme apresentado na Figura 4.11, um dos diferenciais da arquitetura do *SciProv* baseia-se na vinculação do grafo de proveniência em formato RDF para prover uma infraestrutura cujo objetivo final consiste em fornecer recursos para a consulta em SPARQL aos dados de linhagem. SPARQL é a linguagem de consulta padrão recomendada pelo W3C no contexto da *web* semântica.

Ainda de acordo com a Figura 4.11, pode-se observar que tanto o processo de vinculação em RDF quanto a consulta em SPARQL estão associados às ontologias do modelo OPM e às ontologias do domínio referente ao experimento científico modelado. Nesse contexto, a arquitetura do *SciProv* baseia-se no emprego da ontologia em OWL-DL promovida pelo projeto *Open Provenance Model* para a composição do grafo de proveniência serializado em RDF.

Uma vez que a especificação OWL-DL fundamenta-se no campo da Lógica de Descrição [58], é importante observar que a representação em formato RDF obtida como resultado do processo de vinculação descreve os indivíduos que compõem o grafo de proveniência e refere-se, portanto, ao componente *ABox* da ontologia. Por outro lado, o componente *TBox* de uma ontologia tem por objetivo descrever as classes e propriedades. No caso do *SciProv*, o componente *TBox* pode englobar não apenas a representação do modelo OPM quanto do domínio do experimento científico em estudo [59]. A Figura 4.12 ilustra os componentes que descrevem o conhecimento representado no contexto da arquitetura do *SciProv*.

A vinculação do grafo de proveniência em formato RDF no contexto do *SciProv* baseia-se nas diretrizes fornecidas pelo projeto *Open Provenance Model*. A Figura 4.13 apresenta uma visão parcial da hierarquia de classes da ontologia do modelo OPM (*TBox*). Contudo, deve-se observar que tais diretrizes encontram-se em desenvolvimento [54].

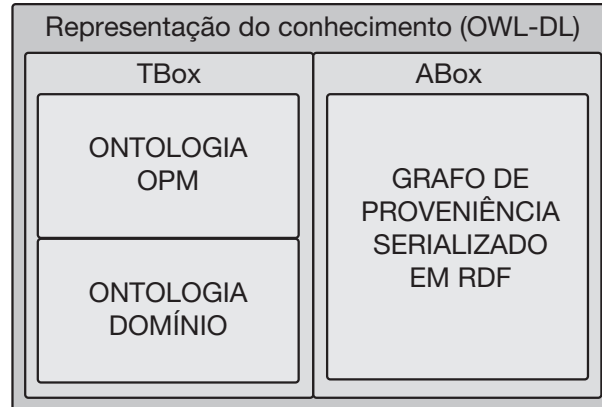


Figura 4.12: Componentes do conhecimento representado na arquitetura do *SciProv*

Além disso, uma vez que a especificação da linguagem RDF/XML impõe uma estrutura sequencial, mecanismos de vinculação podem apresentar restrições para preservar todas as informações contidas em um grafo de proveniência OPM representado na memória. Uma alternativa ao processo de serialização proposto pelo projeto *Open Provenance Model* é apresentado por Zhao [60].

4.1.10 Consulta com Recursos da Web Semântica

A camada de consulta ao grafo de proveniência do *SciProv* (*Provenance Graph Query Layer* na Figura 4.1) tem por objetivo prover um mecanismo e uma interface *web* para a formulação de consultas aos metadados representados em formato da *web* semântica. Nesse contexto, o objetivo da camada de consulta em particular e um dos diferenciais mais importantes da arquitetura do *SciProv* consiste em fornecer recursos oriundos da tecnologia *web* semântica para a consulta à proveniência de dados modelados conforme a especificação OPM.

Coerente com o foco nessa tecnologia, o *SciProv* emprega a linguagem SPARQL para realizar o processamento de consultas aos dados de proveniência utilizando-se máquinas de inferência (*reasoners*). Este recurso da *web* semântica é capaz que efetuar deduções sobre a base de conhecimento do modelo OPM conforme descrito na ontologia específica. Esta abordagem visa obter resultados importantes ao extrair informações adicionais além daquelas que encontram-se expressas de forma direta no grafo de proveniência.

Ainda de acordo com a Figura 4.12, deve-se acrescentar que o processo de consulta semântica aos dados de proveniência referentes à execução de um experimento científico requerem não só o conjunto de asserções coletadas e serializadas em RDF (*ABox*) quanto

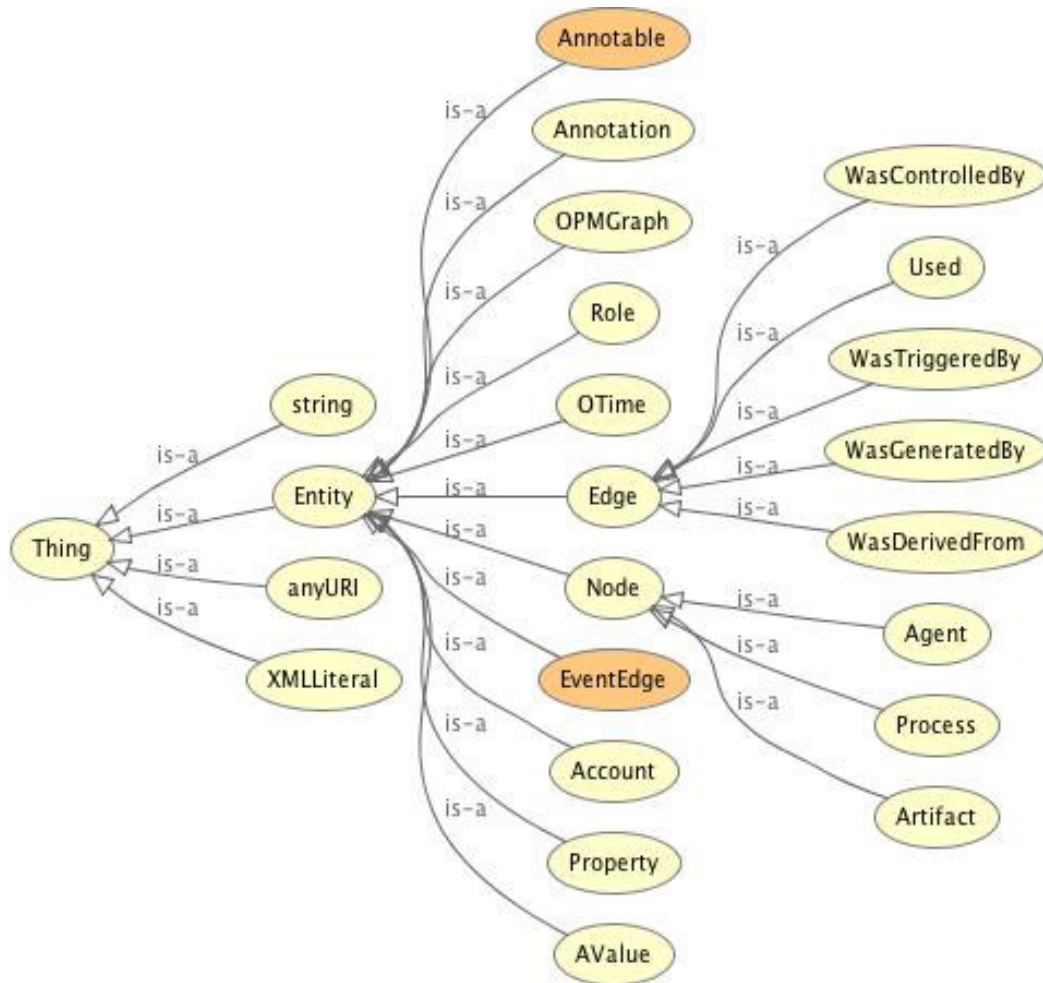


Figura 4.13: Hierarquia de classes da ontologia do OPM

a representação do conhecimento do modelo OPM e do domínio estudado (*TBoxes*).

É importante acrescentar que a arquitetura do *SciProv* prevê o processamento de consultas semânticas a partir de um ou mais *TBoxes* associados ao modelo. Pode-se observar que o requisito mínimo exigido consiste na ontologia do modelo OPM. Nesse contexto, é possível empregar o *TBox* do OPM para formular consultas semânticas aos dados de proveniência (*ABox*) referentes às entidades e dependências causais definidas pelo modelo e processar algumas inferências, conforme apresentado na Seção Regras de Completude 3.5.1. Por exemplo, conforme visto na Figura 3.10, o modelo OPM identifica as regras de completude denominadas “introdução e eliminação de artefatos”. Através de consultas semânticas em SPARQL aos dados de proveniência e ontologia (*TBox* e *ABox*), o *SciProv* pode retornar os resultados processados por inferência.

Porém, a mais importante característica promovida pelo *SciProv* no contexto do processamento semântico dos dados de linhagem baseia-se na ampliação do *TBox* a partir

da concepção e incorporação de especializações ao modelo abstrato de proveniência, cuja relevância é identificada na própria arquitetura em camadas da visão proposta pelo projeto *Open Provenance Model* (ver Figura 4.9). A seção seguinte (4.1.11) apresenta uma discussão sobre essa questão e a relevância no contexto do presente trabalho.

Além disso, é possível vislumbrar ainda que o poder de processamento semântico do *SciProv* pode ser também enriquecido com base na introdução de ontologias que representem conhecimentos específicos relacionadas ao domínio do experimento científico conduzido. Entretanto, o processo de agregação de dois ou mais *TBox* distintos pode requerer o processamento prévio de alinhamento de ontologias, cujo tratamento está além do escopo do presente trabalho.

4.1.11 Especializações ao Modelo OPM

A possibilidade de se construir novos perfis que contenham especializações ao modelo OPM é importante para que novos atributos semânticos possam ser incorporados ao grafo de proveniência. O objetivo consiste em prover um suporte mais abrangente em diversos domínios de aplicações [27]. A Figura 4.14 ilustra as potencialidades que podem ser agregadas ao modelo OPM a partir de novos perfis especializados. O resultado que se espera pode ser representado por consultas semânticas mais ricas e completas aos metadados de linhagem.

A visão proposta pelo projeto *Open Provenance Model* inclui uma camada denominada *OPM Domain Specialization* onde são apontadas áreas de interesse que particularmente podem ser beneficiadas por essa abordagem: *workflows* e aplicações *web* (ver Figura 4.9).

A proposta de um novo perfil deve se caracterizar por formular um ou mais dos seguintes itens ao modelo abstrato, que devem ser incorporadas ao *TBox* da ontologia que representa o OPM estendido:

- Um vocabulário controlado que represente o conhecimento no domínio especializado;
- A definição de subclasses para nós ou arestas do grafo que representem novas subcategorias de entidades ou de relações de dependências causais;
- A definição de propriedades (*object properties* e *data properties*) que expressem novos relacionamentos no contexto do perfil especializado.

A arquitetura do *SciProv* procura identificar uma especialização ao modelo OPM capaz de ampliar o poder de processamento semântico no domínio de *workflows* científicos. Este

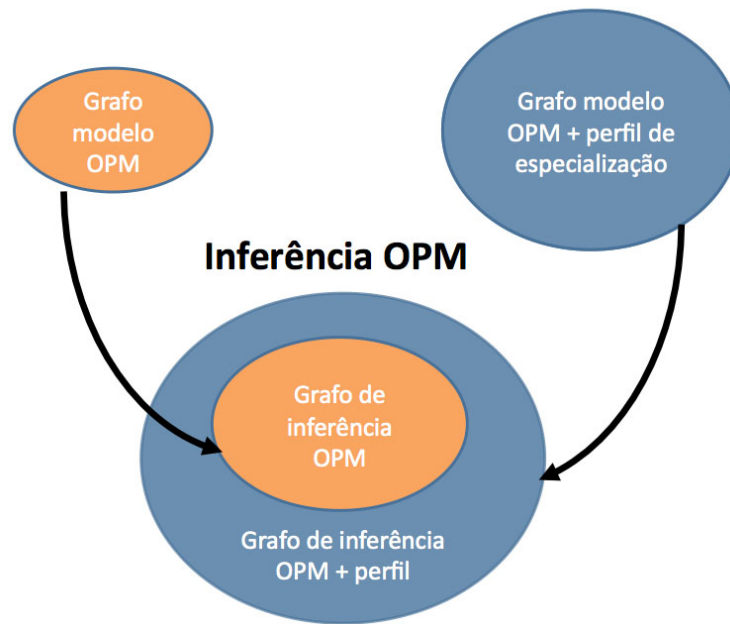


Figura 4.14: Consultas semânticas mais ricas a partir de perfis especializados

recurso constitui uma característica relevante do presente trabalho ao potencializar a capacidade de inferência sobre os metadados de proveniência coletados em um cenário de *e-Science*.

Uma das propostas incluídas na especialização apresentada está relacionada às propriedades (*object properties*) denominadas *derivedArtifact* e *derivedFromArtifact* da ontologia OWL-DL do modelo OPM. Tais propriedades representam relacionamentos entre as classes *Derived* e *Artifact*.

A Figura 4.15 exibe a propriedade *derivedFromArtifact*, que liga indivíduos da classe *Derived* (domínio) a indivíduos da classe *Artifact* (alcance ou *range*). Tal propriedade tem por objetivo relacionar as entidades OPM do tipo artefato que foram geradas a partir de uma relação de dependência causal do tipo *was derived from*.

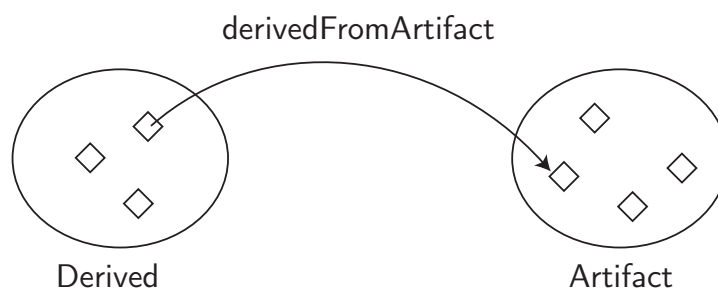


Figura 4.15: Propriedade *derivedFromArtifact* da ontologia do OPM

Segundo a representação do conhecimento modelado na ontologia do OPM, as propriedades *derivedArtifact* e *derivedFromArtifact* não são consideradas inversas e transitivas. Considere-se um caso hipotético (Figura 4.16) onde é apresentado um grafo de proveniência OPM contendo arestas que representam relações de dependência causal do tipo *was derived from*. Nesse cenário, o modelo OPM unicamente declara os três relacionamentos identificados por linhas contínuas na Figura 4.16.

Em um modelo especializado no domínio de *workflows* científicos, pode-se considerar as relações de inversibilidade e de transitividade para as propriedades *derivedArtifact* e *derivedFromArtifact*. Assim, conforme o exemplo hipotético ilustrado na Figura 4.16, caso o artefato *A3* tenha sido derivado do artefato *A2* e o artefato *A2* por sua vez tenha sido derivado do artefato *A1*, as características propostas para *derivedFromArtifact* permitem o processamento de inferências identificadas por linhas pontilhadas na Figura 4.16, que expressam relações de dependência causal do tipo *was derived from* inferidas entre os diversos artefatos representados.

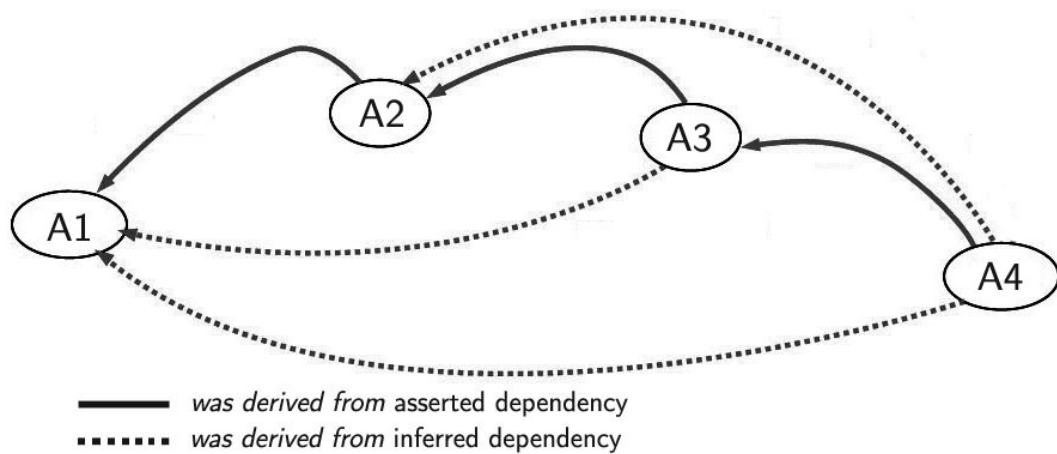


Figura 4.16: Processamento semântico a partir da especialização do modelo OPM

Em um contexto de proveniência de *workflows* científicos, estas propriedades configuradas como inversas e transitivas implicam dizer que é possível estabelecer um relacionamento de dependência causal do tipo *was derived from* ao longo do processo de transformação de um dado desde a sua origem até o resultado final obtido computacionalmente. Pode-se argumentar que tal refinamento do modelo OPM é relevante para a construção de um grafo de proveniência semanticamente mais rico e abrangente no escopo do presente trabalho.

Ainda no contexto da proveniência de dados e processos em *workflows* científicos,

é possível estabelecer outras extensões semelhantes para o modelo abstrato OPM que podem compor um perfil específico para o domínio estudado. Por exemplo, pode-se considerar que a relação de dependência causal *was triggered by* entre processos — entende-se aqui um *processo* ou atividade em um *workflow* como um tipo de artefato OPM — seja transitiva, de forma a estabelecer um mecanismo capaz de processar inferências quanto à linhagem dos processos executados em um dado experimento científico modelado computacionalmente.

Para finalizar esta seção, vale mencionar que uma outra importante possibilidade de ampliação da capacidade de processamento semântico do modelo OPM no domínio de *workflows* científicos refere-se ao arcabouço de anotações (Seção 3.6.1). Nesse cenário, tanto pode-se vislumbrar o emprego de anotações como um mecanismo para prover uma maior interoperabilidade entre sistemas de proveniência distintos quanto para a ampliação do conteúdo semântico referente a um domínio estudado.

Nessa última hipótese, extensões ao arcabouço de anotações podem ser propostas para permitir a formulação de consultas semânticas ainda mais ricas e complexas aos dados de proveniência modelados segundo o *Open Provenance Model* no contexto estudado. É importante observar que a arquitetura do *SciProv* apresentada no presente trabalho permite o processamento de inferências com base no arcabouço de anotações OPM. No Capítulo 5 é apresentado um caso de uso específico para o tratamento de anotações.

4.2 Implementação da Arquitetura do *SciProv*

A implementação da arquitetura do *SciProv* baseia-se no emprego de um conjunto de tecnologias oriundas da Ciência da Computação onde a ênfase reside no emprego de *software* livre e de código aberto.

Pode-se destacar o emprego das seguintes tecnologias no desenvolvimento da arquitetura do *SciProv*:

- *Linguagem de Programação Java*¹.
- Sistema de Gerenciamento de Banco de Dados Relacional *Apache Derby*².
- Servidor *web* e de aplicações *open source GlassFish*³.

¹Java disponível em <http://www.sun.com/java/>

²Apache Derby disponível em <http://db.apache.org/derby/>

³GlassFish disponível em <https://glassfish.dev.java.net/>

- Ambiente de desenvolvimento integrado *Netbeans*⁴.

Concretamente, o *SciProv* consiste em um sistema baseado na *web* desenvolvido para execução a partir de qualquer navegador. Essa característica é relevante no contexto da arquitetura, que tem por objetivo processar informações em um ambiente colaborativo de pesquisa porém distribuído e interconectado a partir de uma grade computacional.

A Figura 4.17 apresenta a interface gráfica da página inicial do *SciProv* visualizada a partir do navegador *web* Mozilla Firefox no ambiente operacional Windows XP.

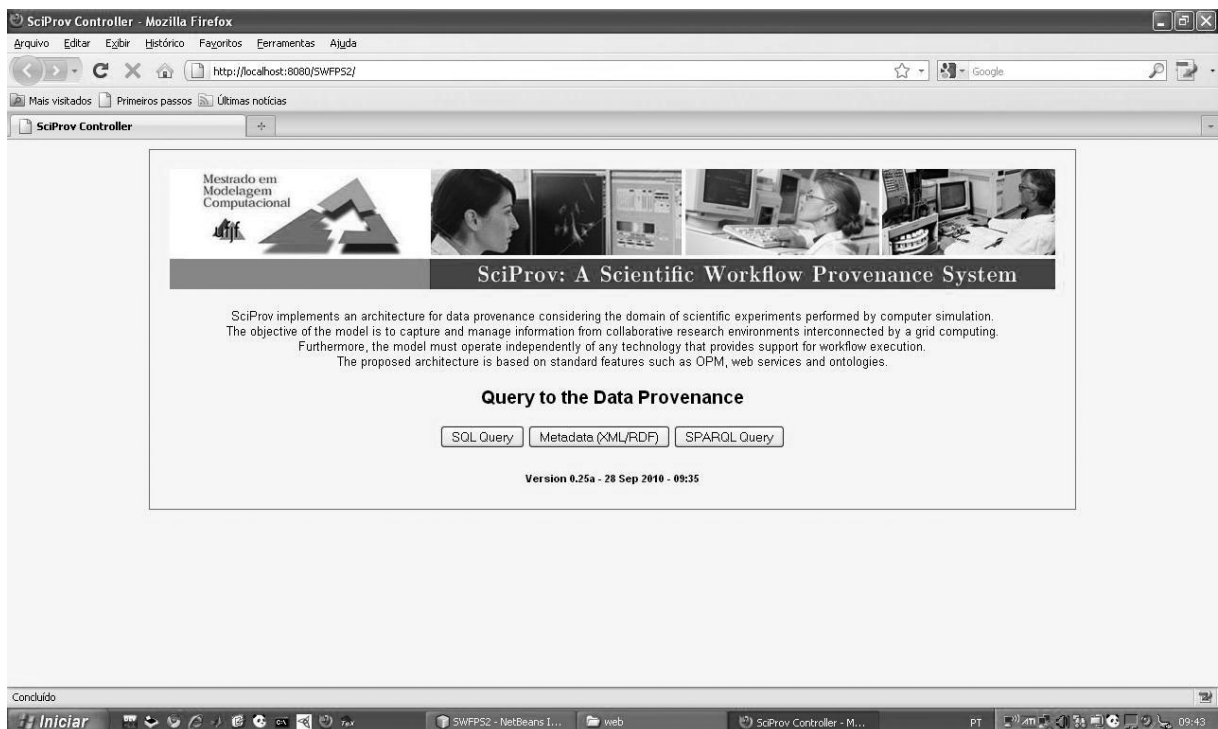


Figura 4.17: Página inicial do *SciProv*

4.2.1 Implementação do Mecanismo de Instrumentalização

A implementação do mecanismo de instrumentalização do *SciProv* teve como premissa o suporte a SGWfCs relevantes no contexto de *e-Science*. Os SGWfCs adotados para a implementação de protótipos e avaliação de resultados obtidos foram Kepler, Taverna e Vistrails.

Para efeito de ilustração pode-se considerar um *workflow* científico conforme apresentado na Figura 4.18, modelado no SGWfC Kepler. É importante observar que o *workflow* analisado a partir desta seção é composto apenas dos componentes básicos necessários ao

⁴Netbeans disponível em <http://netbeans.org/>

entendimento do mecanismo de instrumentalização e geração do grafo de proveniência. O capítulo 5 discute a aplicação do *SciProv* em um conjunto de estudos de caso relacionados à Bioinformática.

No *workflow* apresentado na Figura 4.18, o componente denominado *Add1ToListBlack* corresponde a artefato de *software* encapsulado na forma de um serviço *web* cujo objetivo é receber uma lista de inteiros como parâmetro de entrada e disponibilizar uma outra lista de inteiros como argumento de saída. O processamento do serviço *web* consiste unicamente em adicionar uma unidade a cada elemento da lista de inteiros.

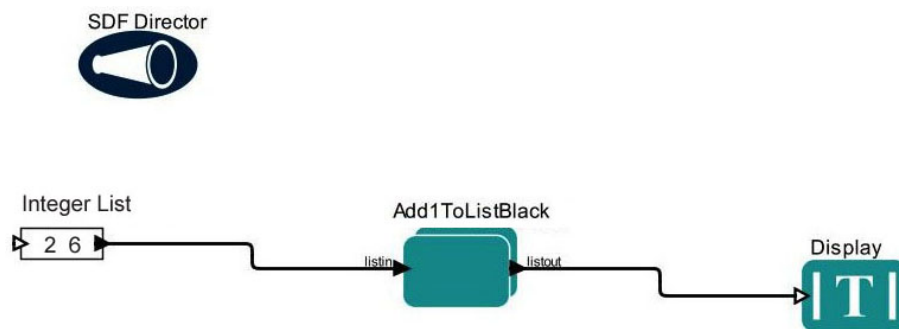


Figura 4.18: *Workflow* científico modelado no SGWfC Kepler

No contexto do *SciProv*, o processo de instrumentalização deve ser realizado manualmente a partir do próprio SGWfC utilizado pelo pesquisador ou cientista para a orquestração do *workflow* científico. Em um ambiente de pesquisa colaborativo pode-se considerar que os experimentos modelados computacionalmente poderão ser acessados no sentido de prover o mecanismo de instrumentalização necessário para a captura dos dados de proveniência.

A Figura 4.19 apresenta o processo de instrumentalização manual referente ao *workflow* descrito na Figura 4.18.

A instrumentalização apresentada na Figura 4.19 se dá a partir da adaptação manual do componente *Add1ToListBlack* conforme discutido na Seção 4.1.3. Como resultado do processo de instrumentalização, é gerado um componente composto que abstrai do usuário o mecanismo de coleta dos dados de proveniência incluído no *workflow* científico. Como resultado, a apresentação do *workflow* na interface gráfica do SGWfC pode se manter praticamente inalterada para o pesquisador. Deve-se mencionar que o conceito de componente composto encontra-se disponível em SGWfCs como Kepler, Taverna e Vistrails.

- Identificador da aresta do grafo de proveniência (relação de dependência causal) coletada pela instância do serviço *web* instrumentalizador.
- Identificador do nó (Artefato, Processo ou Agente) do grafo de proveniência relacionado através da dependência causal definida no item anterior.

O segundo argumento de entrada para o serviço *web* de instrumentalização tem por objetivo capturar as informações de entrada ou de saída do componente original do *workflow*, dependendo da posição onde o componente de instrumentalização é inserido no fluxo de dados.

Conforme mencionado, o *workflow* modelado no SGWfC Kepler apresentado na Figura 4.19 emprega o serviço *web* de instrumentalização para coletar os dados de proveniência quatro vezes. As instâncias do componente de instrumentalização denominadas *Process*, *Artifact1* e *Artifact2* referem-se aos respectivos nós do grafo de proveniência OPM para uma descrição (*account*) chamada *black*.

As instâncias do componente de instrumentalização referentes aos artefatos também capturam as dependências causais do tipo *used* e *was generated by* entre os nós identificados no *workflow*. Por sua vez, a instância do componente de instrumentalização referente ao processo (*Process*) captura a dependência causal do tipo *was controlled by* entre este nó do grafo e o nó da entidade agente.

A Figura 4.20 apresenta o grafo de proveniência segundo o modelo OPM gerado pelo *SciProv* para uma execução específica de um *workflow* conforme apresentado na Figura 4.19, onde à lista de inteiros (2,6) é adicionado 1 a cada um dos elementos, gerando como resultado a lista (3,7).

4.2.2 Regra de Completude Inválida

Ainda a partir da Figura 4.19, a instância do componente de instrumentalização denominada *WasDerivedFrom* captura a dependência causal do tipo *was derived from* entre os artefatos existentes no *workflow*. É importante observar que, segundo o modelo abstrato OPM, as dependências causais do tipo *was derived from* devem ser explicitamente declaradas. Em outros termos, o OPM não considera uma regra de completude válida a asserção ilustrada na Figura 4.21.

Portanto, segundo o OPM, dado que o artefato *A2* foi gerado pelo processo *P1* e que o processo *P1* usou o artefato *A1*, a asserção que declara que o artefato *A2* foi derivado

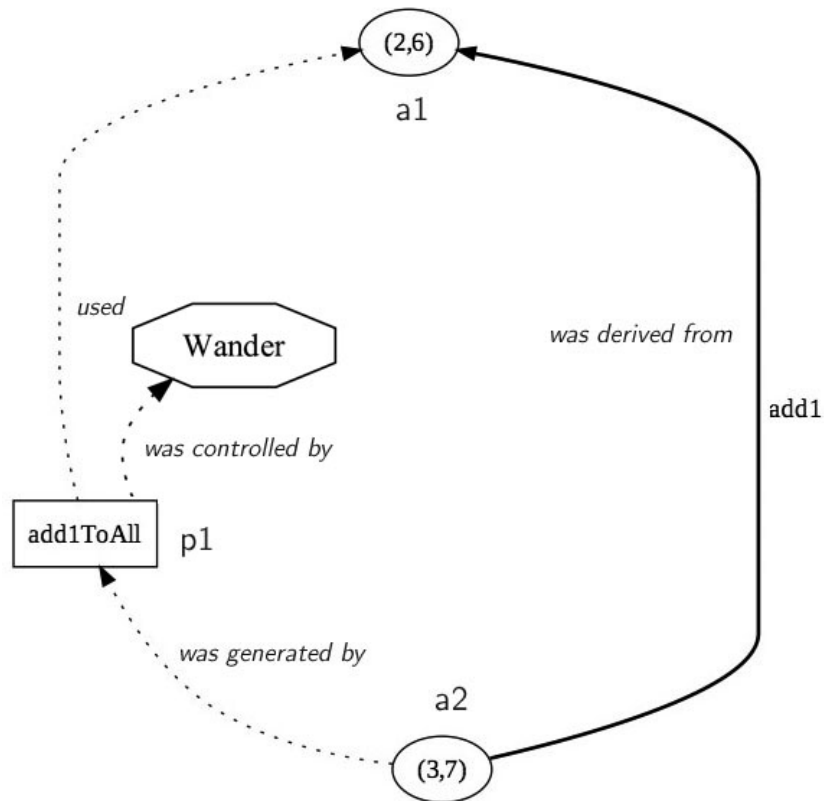


Figura 4.20: Grafo de Proveniência gerado no *SciProv*

do artefato $A1$ representa uma regra de completude inválida porque o modelo não pode garantir que existe uma dependência de dados entre $A2$ e $A1$ originada a partir da execução do processo $P1$.

Entretanto, conforme apresentado na Seção 4.1.11, o OPM contempla a possibilidade de se construir perfis para promover especializações ao modelo padrão de forma a incorporar novos atributos semânticos ao grafo de proveniência. O objetivo consiste em fornecer um suporte mais abrangente em domínios específicos como *workflows* científicos [27].

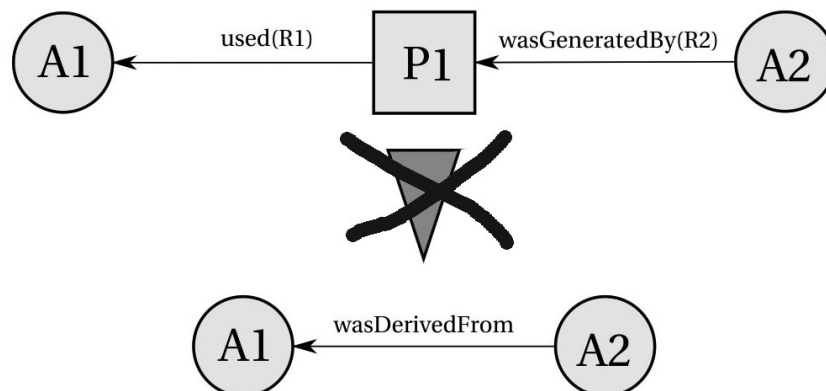


Figura 4.21: Regra de Completude inválida no OPM

Assim, poderia argumentar-se que uma asserção como a “regra de completude” pode ser válida, considerando-se subclasses — em uma ontologia OWL — de entidades OPM como artefatos e processos e de subpropriedades como *was derived from* em um perfil especializado no contexto de *e-Science*.

Nesse contexto, a arquitetura do *SciProv* procura formular refinamentos para o *TBox* do OPM que representem o conhecimento no domínio de *workflows* científicos e que podem compor um perfil ao modelo. A implementação apresentada nesse capítulo ilustra na Seção 4.2.10 este importante recurso incorporado ao presente trabalho.

4.2.3 Instrumentalização em Dois Níveis de Detalhamento

Tomando-se como base o exemplo de *workflow* modelado no SGWfC Kepler apresentado na Figura 4.22, o mecanismo de instrumentalização construído manualmente exibido na Figura 4.23 pode ser considerado como uma captura de proveniência de acordo com o modelo OPM em um nível de granularidade específico, definido de acordo com os interesses do pesquisador ou grupo de pesquisa envolvido com o experimento científico em estudo.

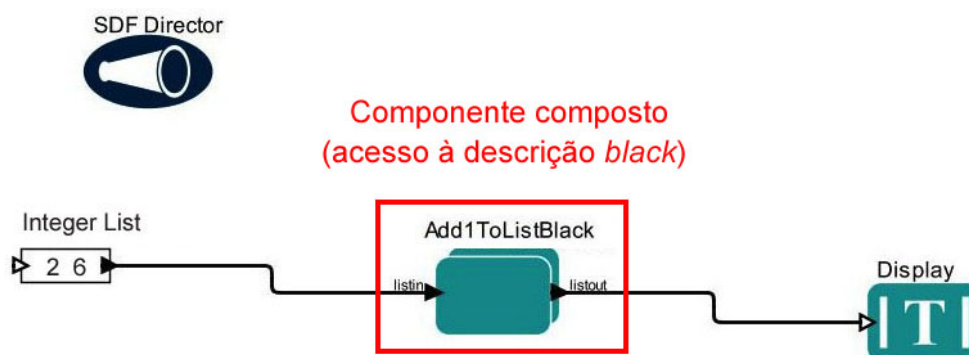


Figura 4.22: Componente composto modelado no SGWfC Kepler

Conforme é possível observar a partir de cada uma das cadeias de caracteres que representa o primeiro argumento de entrada para as instâncias do serviço *web* de instrumentalização (Figura 4.23), uma descrição (*account*) denominada *black* é especificada para os dados de proveniência coletados. Como consequência, todos os dados coletados a partir dessa instrumentalização referem-se à descrição denominada *black*.

Caso o *workflow* científico seja instrumentalizado em mais de um nível de granularidade a partir de descrições distintas, o subgrafo referente a cada *account* é exibido em uma cor diferente no grafo de proveniência gerado. Por uma questão de analogia, o subgrafo que

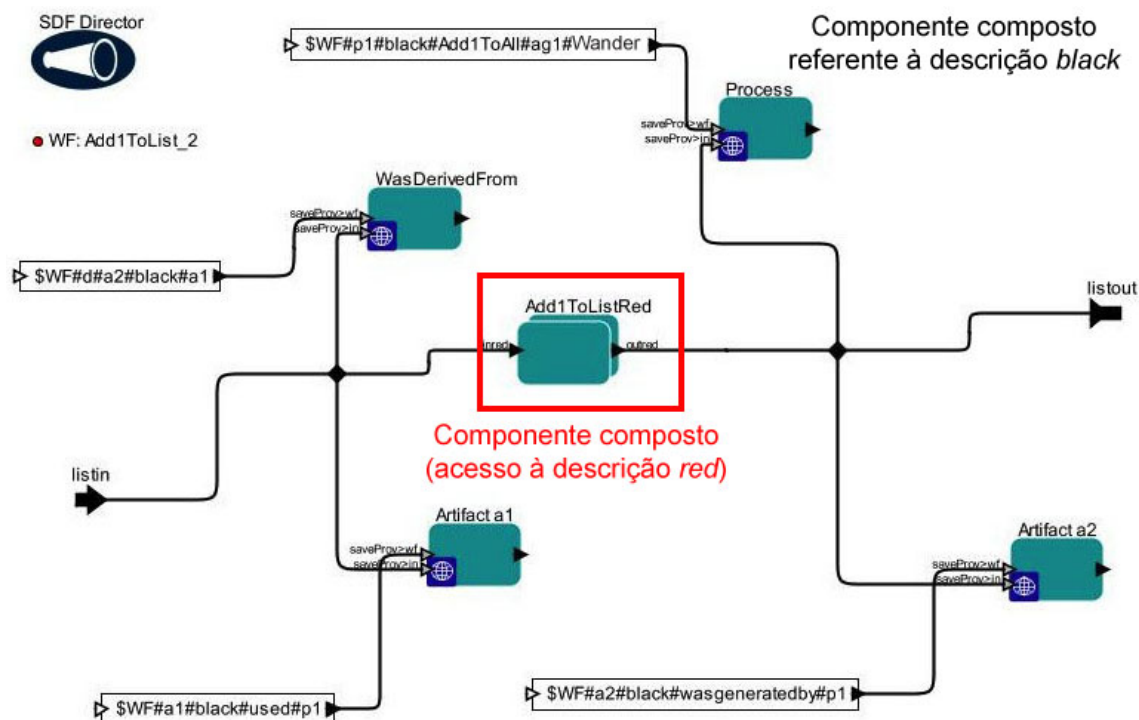


Figura 4.23: Instrumentalização de componente composto – descrição *black*

descreve a proveniência de dados — incluindo-se arestas e nós — da descrição *black* é exibido na cor preta (Figura 4.24).

A questão de maior relevância relaciona-se à possibilidade de instrumentalização do *workflow* em diversos níveis de refinamento. Nesse contexto, a partir do emprego de componentes compostos do SGWfC Kepler — recurso também disponível no Taverna e Vistrails —, é possível construir uma segunda descrição para o *workflow* instrumentalizado, conforme representado na Figura 4.24.

Nessa descrição, denominada *red*, ocorre um processo de instrumentalização em um nível mais fino de granularidade. Por consequência, é possível coletar e posteriormente consultar os dados de proveniência em um grau de detalhamento maior em relação à descrição *black*.

Os dados coletados durante a execução do *workflow* e associados à descrição *red* são representados na cor vermelha no grafo de proveniência gerado (Figura 4.25).

Pode-se observar que embora tanto a descrição *black* quanto a descrição *red* referem-se a uma mesma execução de um *workflow* científico, a semântica representada a partir do grafo de proveniência apresenta diferenciações. Portanto, a partir da coleta de dados em níveis diversos de granularidade é possível proceder análises e consultas de proveniência

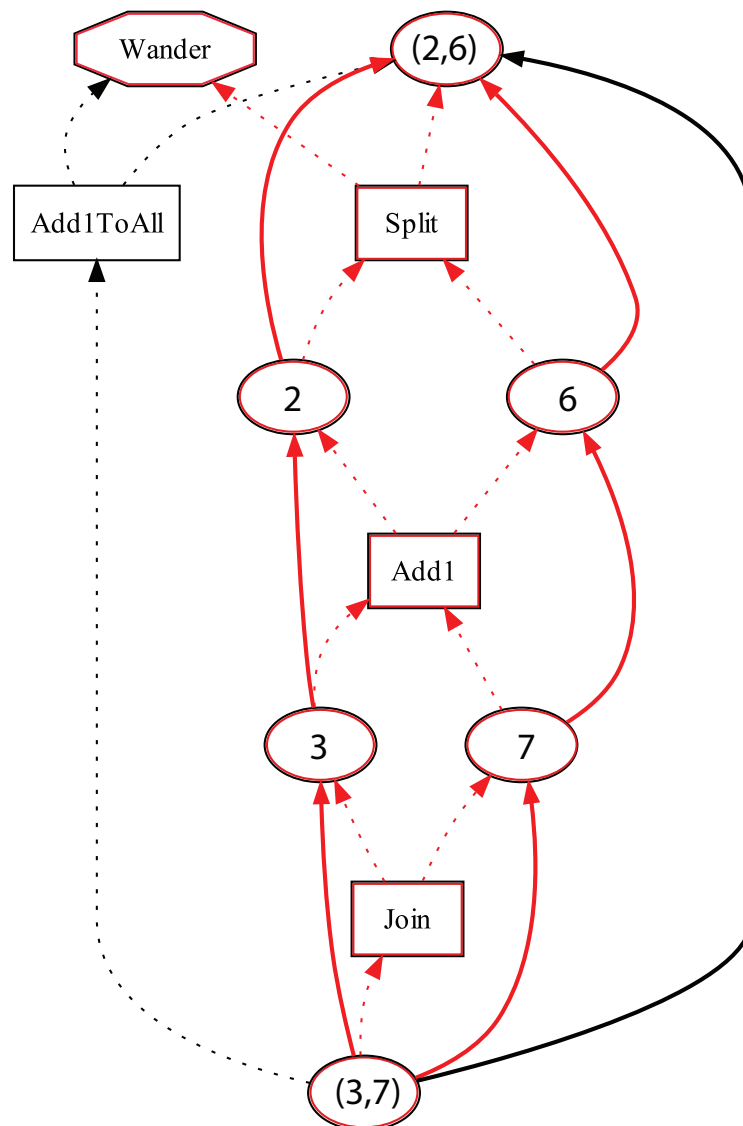


Figura 4.24: Grafo de Proveniência OPM com descrições distintas

distintas, de acordo com as particularidades dos estudos em andamento desenvolvidos pelo pesquisador.

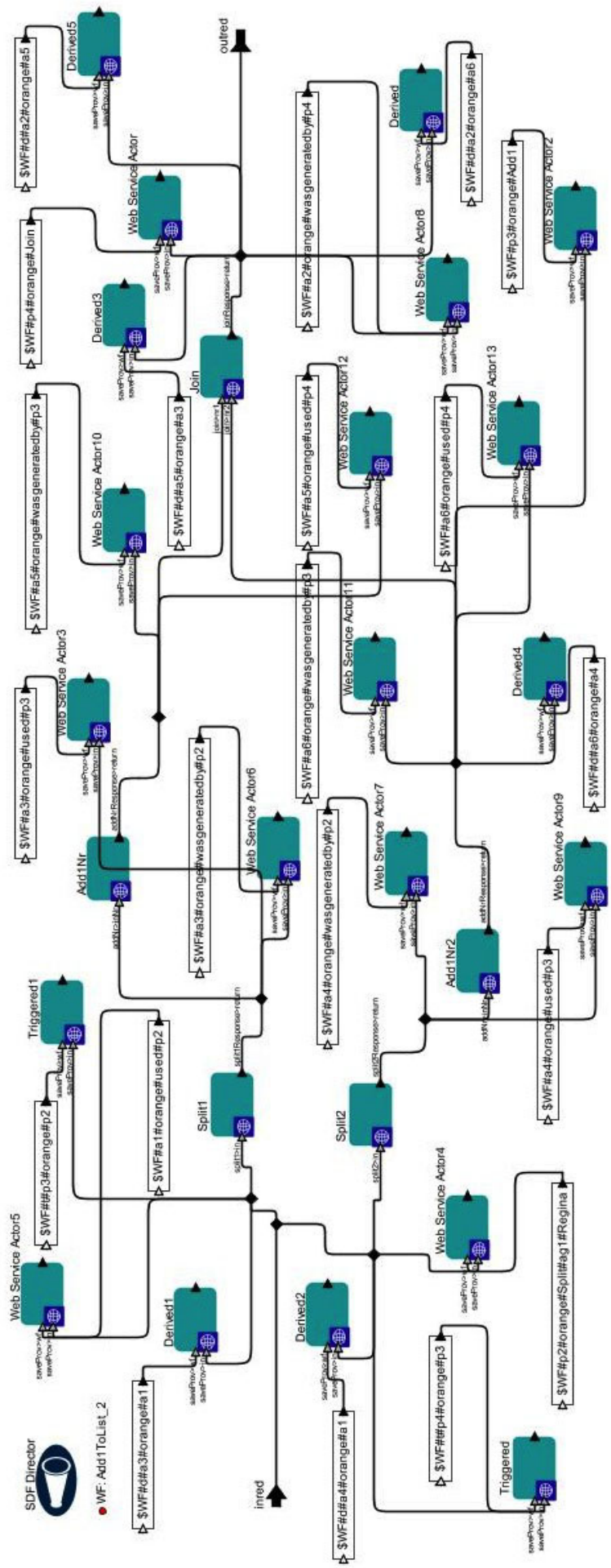


Figura 4.25: Instrumentalização em nível fino de detalhamento – descrição red

4.2.4 Implementação da Persistência Relacional

O *SciProv* emprega uma base de dados relacional para prover a persistência dos dados de proveniência coletados. O protótipo construído utiliza especificamente o SGBDR *Apache Derby*⁵ para o armazenamento dos dados de linhagem capturados segundo o modelo OPM. O *Apache Derby* é um banco de dados relacional escrito em Java e desenvolvido como um projeto *open source*.

A Figura 4.26 apresenta algumas tuplas da tabela *Causal_Dependency* conforme implementada em um protótipo construído durante a implementação da arquitetura do *SciProv*.

#	WORKFLOW_REF	CAUSAL_DEPENDENCY_ID	EFFECT_REF	ROLE_VALUE	CAUSE_REF	ACCOUNT_REF	TIME_NOLATERTHAN	TIME_NOEARLIERTHAN
1	Add1ToList_1	wastriggeredby	p4	was triggered by	p3	orange	2010-08-04 13:27:42.328	2010-08-04 13:27:42.328
2	Add1ToList_1	wastriggeredby	p3	was triggered by	p2	orange	2010-08-04 13:27:42.437	2010-08-04 13:27:42.437
3	Add1ToList_1	wasderivedfrom	a2	was derived from	a6	orange	2010-08-04 13:27:42.5	2010-08-04 13:27:42.5
4	Add1ToList_1	wasderivedfrom	a2	was derived from	a5	orange	2010-08-04 13:27:42.562	2010-08-04 13:27:42.562
5	Add1ToList_1	wasderivedfrom	a6	was derived from	a4	orange	2010-08-04 13:27:43.062	2010-08-04 13:27:43.062
6	Add1ToList_1	wasderivedfrom	a5	was derived from	a3	orange	2010-08-04 13:27:43.125	2010-08-04 13:27:43.125
7	Add1ToList_1	wasderivedfrom	a3	was derived from	a1	orange	2010-08-04 13:27:43.187	2010-08-04 13:27:43.187
8	Add1ToList_1	wasderivedfrom	a4	was derived from	a1	orange	2010-08-04 13:27:43.25	2010-08-04 13:27:43.25
9	Add1ToList_1	used	p4	used	a6	orange	2010-08-04 13:27:43.343	2010-08-04 13:27:43.343
10	Add1ToList_1	used	p4	used	a5	orange	2010-08-04 13:27:43.421	2010-08-04 13:27:43.421
11	Add1ToList_1	wasgeneratedby	a6	was generated by	p3	orange	2010-08-04 13:27:43.484	2010-08-04 13:27:43.484
12	Add1ToList_1	wasgeneratedby	a5	was generated by	p3	orange	2010-08-04 13:27:43.562	2010-08-04 13:27:43.562
13	Add1ToList_1	used	p3	used	a4	orange	2010-08-04 13:27:43.64	2010-08-04 13:27:43.64

Figura 4.26: Tuplas da tabela *Causal_Dependency* do *SciProv*

Note-se que o campo *role_value* é preenchido com informação idêntica ao campo *causal_dependency_id*. Nesse exemplo, não foi definido um papel específico para cada aresta do grafo de proveniência e, nesse caso, a informação padrão persistida para o campo é o tipo de dependência causal.

Pode-se observar também que, segundo o modelo OPM, as dependências causais que aparecem nas tuplas apresentadas na Figura 4.26 fazem referência a apenas uma observação de tempo. Nesse caso, o protótipo implementado para o *SciProv* persiste a mesma observação do tempo nos campos *Time_NoLaterThan* e *Time_NoEarlierThan*.

Uma vez que as informações de proveniência coletadas são armazenadas em uma base de dados relacional, a arquitetura do *SciProv* oferece uma interface com o usuário para a consulta aos dados de linhagem a partir da linguagem padrão SQL (Figura 4.27).

A consulta apresentada na Figura 4.27 tem por objetivo recuperar todas as dependências causais do tipo *wasderivedBy* que possuem o valor *a2* para o campo *effect_ref* referente ao *workflow* nomeado *Add1ToList_3*. O resultado da consulta SQL a partir da

⁵Apache Derby disponível em <http://db.apache.org/derby/>

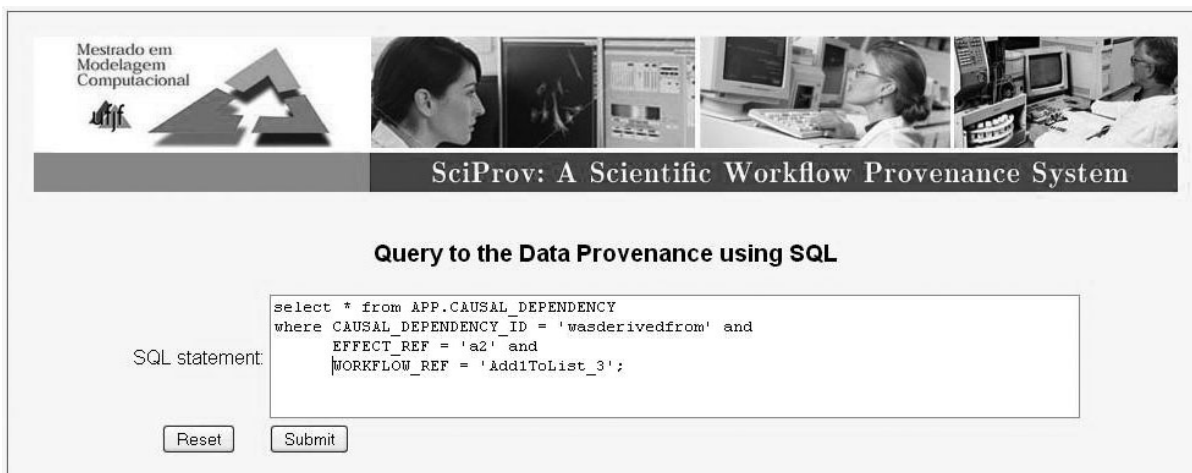


Figura 4.27: Interface gráfica do *SciProv* para consultas SQL

interface com o usuário do *SciProv* é apresentado na Figura 4.28.

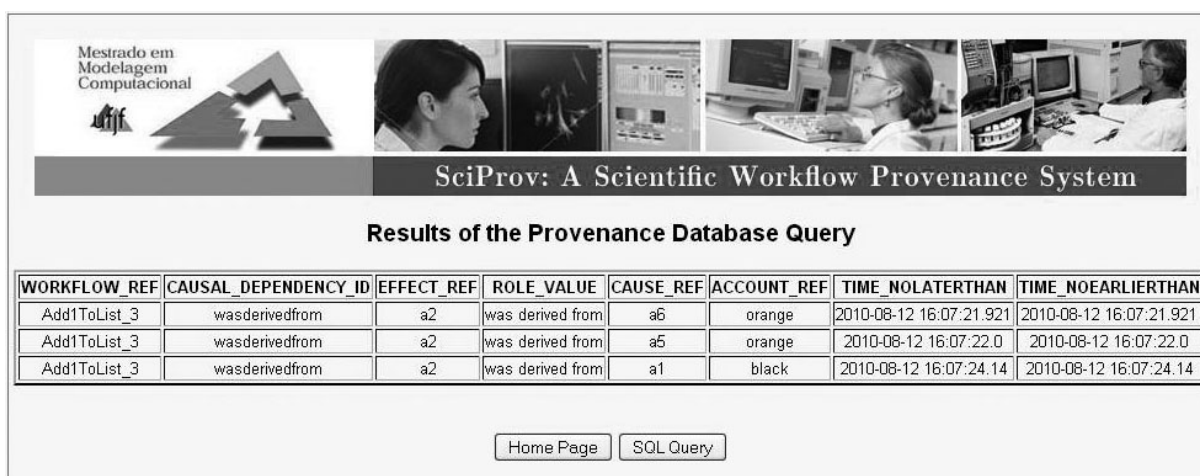


Figura 4.28: Resultado da consulta SQL na interface com o usuário do *SciProv*

4.2.5 Implementação do Grafo de Proveniência

Com o objetivo de facilitar a adoção e implementação do modelo abstrato OPM, o projeto Open Provenance Model preconiza o emprego de um conjunto de APIs e bibliotecas, *middleware* e guias de estilo segmentados em camadas que se inter-relacionam com o núcleo do modelo (ver Figura 4.8):

- Tutoriais e exemplos de aplicação.
- Esquemas XML e RDF e ontologia OWL-DL do modelo OPM.
- Especificação 1.1 do modelo OPM.

- “Caixa de Ferramentas” *Toolbox* com aplicações de linha de comando e APIs de classes Java para a manipulação de grafos de proveniência OPM.

É importante observar que a implementação do *SciProv* utiliza de forma abrangente as diretrizes promovidas pelo projeto *Open Provenance Model*. Esta característica constitui um dos aspectos relevantes no contexto do presente trabalho. Esta abordagem tem por objetivo manter o desenvolvimento da arquitetura aderente às especificações do modelo OPM. Além disso, o emprego de classes Java, esquemas em linguagem de descrição XML e XML/RDF e ontologias em formato OWL encontram-se em consonância com a diretriz do trabalho no que concerne à utilização de soluções baseadas em *software* livre e em padrões endossados e recomendados pelo W3C.

No contexto da arquitetura do *SciProv*, um dos objetivos da camada de geração e visualização do grafo de proveniência (*Provenance Graph Generation and Visualization Layer* na Figura 4.1) consiste em processar os dados de um determinado *workflow* científico persistidos na base de dados relacional e produzir uma representação em memória aderente ao modelo de proveniência OPM.

Na implementação da arquitetura torna-se necessário que o usuário selecione a partir da interface *web* do *SciProv* o nome e número da execução que identificam o *workflow* cujos dados de proveniência persistidos deverão ser modelados em memória, conforme apresentado na Figura 4.29.



Figura 4.29: Interface com o usuário para seleção do *workflow* a ser representado na memória

O *SciProv* emprega a implementação do modelo OPM no *middleware* de conteúdo semântico *Tupelo*⁶ para construir uma representação em memória do grafo de proveniência.

⁶OPM em Tupelo disponível em <http://tupeloproject.ncsa.uiuc.edu/node/2>

O objetivo do *middleware* Tupelo é promover a interoperabilidade ao implementar um sistema de gerenciamento de dados e metadados baseado em tecnologia *web* semântica e idéias de *Content Management Systems* (CMS). Usando essa abstração, Tupelo fornece APIs para leitura, escrita, e consulta a dados e metadados a partir de diversos sistemas de armazenamento e protocolos existentes, incluindo sistemas de arquivos, bancos de dados relacionais e semiestruturados. Tupelo fornece ainda uma série de APIs e bibliotecas, de forma que aplicações Java possam utilizar os recursos do *middleware* [61].

Tupelo baseia-se em uma abstração denominada “contexto”, que representa uma espécie de visão semântica dos recursos manipulados pelo *middleware* [61]. A partir de um contexto é possível criar instâncias de classes Java para as diversas entidades OPM persistidas na base relacional — artefatos, processos, agentes, dependências causais, etc. — e construir um grafo que represente os relacionamentos de dependência existentes. Como resultado desse processo, obtém-se uma representação em memória para o grafo de proveniência.

4.2.6 Serialização em Formatos XML e DOT

A partir da implementação de um contexto do Tupelo que modela o grafo de proveniência OPM em memória, é possível promover a vinculação em formato XML. Para que esse processo seja possível, a representação do modelo OPM no *middleware* inclui o esquema XML que contém os espaços de nomes utilizados na serialização.

A Figura 4.30 apresenta um exemplo de arquivo em formato XML obtido como resultado do processo de serialização do grafo de proveniência gerado em memória pelo *SciProv*. O mecanismo de vinculação emprega a implementação disponível no Tupelo referente à versão 1.1 do modelo OPM.

A partir da representação em formato XML do grafo de proveniência OPM, o *SciProv* permite implementar a serialização em formato DOT [57]. No contexto do *SciProv*, o grafo de proveniência OPM serializado em XML conforme apresentado na Figura 4.30, após convertido no formato da linguagem DOT, gera a representação gráfica em formato *Portable Document File* (PDF) exibida na Figura 4.31.

Cabe observar que o *SciProv* oferece o recurso de exibição tanto do arquivo XML serializado quanto da visualização gráfica do grafo de proveniência a partir da interface com o usuário.



```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opm:opmGraph xmlns:opm="http://openprovenance.org/model/v1.1.a" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns="http://example.com/">
  <opm:accounts>
    <opm:account id="orange"/>
    <opm:account id="black"/>
    <opm:overlaps>
      <opm:account ref="orange"/>
      <opm:account ref="black"/>
    </opm:overlaps>
  </opm:accounts>
  <opm:processes>
    <opm:process id="p3">
      <opm:account ref="orange"/>
      <opm:label value="Add1"/>
    </opm:process>
    <opm:process id="p4"/>
    <opm:process id="p2"/>
    <opm:process id="p1"/>
  </opm:processes>
  <opm:artifacts>
    <opm:artifact id="a6">
      <opm:account ref="orange"/>
      <opm:label value="1"/>
    </opm:artifact>
    <opm:artifact id="a5"/>
    <opm:artifact id="a4"/>
    <opm:artifact id="a3"/>
    <opm:artifact id="a2"/>
    <opm:artifact id="a1"/>
    <opm:artifact id="a2"/>
    <opm:artifact id="a1"/>
  </opm:artifacts>
  <opm:agents/>
  <opm:causalDependencies/>
</opm:opmGraph>

```

Figura 4.30: Serialização em formato XML do grafo de proveniência OPM

4.2.7 Serialização em Formato RDF

O *SciProv* utiliza a representação em memória do grafo de proveniência para promover a serialização em RDF. Para processar a vinculação, a arquitetura emprega a infraestrutura disponibilizada a partir do *middleware* Tupelo.

Conforme apresentado na Seção 4.1.9 (Figura 4.12), a representação do conhecimento a partir de uma ontologia descrita em formato OWL-DL baseia-se na Lógica de Descrição cujos componentes são referenciados como *TBox* e *ABox*. Nesse contexto, a arquitetura do *SciProv* emprega o *TBox* do modelo OPM proposto pelo projeto *Open Provenance Model*. A Figura 4.32 apresenta o componente *TBox* da ontologia do modelo OPM especificada em OWL-DL.

Tupelo emprega o conceito de Identificador Uniforme de Recursos (do inglês *Uniform Resource Identifier* – URI) para a identificação global dos metadados e semântica explícita de modo que as informações criadas e gerenciadas através do *middleware* possam ser

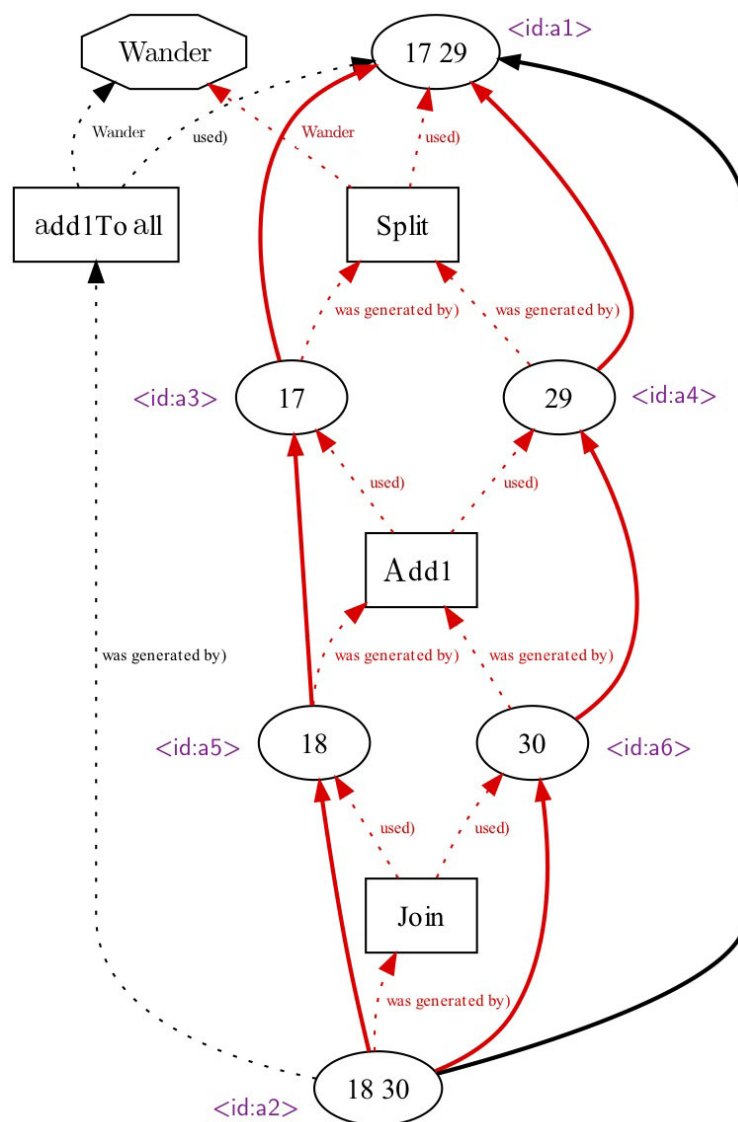


Figura 4.31: Representação visual do grafo de proveniência OPM

exportadas e utilizadas por aplicações baseadas em RDF, como é o caso do *SciProv*. Pode-se observar na Figura 4.33 as referências as URIs definidas no contexto do Tupelo para os metadados definidos segundo a ontologia do modelo OPM.

É importante observar que a implementação do modelo OPM no contexto do *middleware* Tupelo emprega a infraestrutura fornecida pelo *Sesame*, um arcabouço de código aberto no contexto da *web* semântica para armazenamento, inferência e consulta a metadados em RDF [62]. Em outros termos, Tupelo emprega APIs do arcabouço *Sesame* para prover a serialização em formato RDF do grafo de proveniência. O arquivo gerado pode ser interpretado como o componente *ABox* no contexto da representação semântica em OWL-DL das informações de linhagem coletadas.

```

<?xml version="1.0"?>
<rdf:RDF
  xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
  xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
  xmlns:swrl="http://www.w3.org/2003/11/swrl#"
  xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:opm="http://www.ipaw.info/2007/opm#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xml:base="http://www.ipaw.info/2007/opm">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="OpmEntity"/>
  <owl:Class rdf:ID="Node">...</owl:Class>
  <owl:Class rdf:ID="Controlled">...</owl:Class>
  <owl:Class rdf:ID="Used">...</owl:Class>
  <owl:Class rdf:ID="Account">...</owl:Class>
  <owl:Class rdf:ID="TimeInterval">...</owl:Class>
  <owl:Class rdf:ID="Generated">...</owl:Class>
  <owl:Class rdf:about="#Event">...</owl:Class>
  <owl:Class rdf:ID="Triggered">...</owl:Class>
  <owl:Class rdf:ID="Derived">...</owl:Class>
  <owl:Class rdf:ID="Agent">...</owl:Class>
  <owl:Class rdf:ID="Artifact">...</owl:Class>
  <owl:Class rdf:ID="Role">...</owl:Class>
  <owl:Class rdf:ID="Process">...</owl:Class>
  <owl:ObjectProperty rdf:ID="eventRole">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="controlledStartTime">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="usedArtifact">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="derivedFromArtifact">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="usedTime">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="controlledAccount">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="triggeredTime">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="controlledProcess">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="generatedRole">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#eventAccount">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="usedAccount">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="triggeredByProcess">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="derivedArtifact">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="usedRole">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#eventArtifact">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="usedByProcess">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="derivedTime">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="controlledTerminateTime">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="generatedByProcess">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="triggeredProcess">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="derivedAccount">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="controlledByAgent">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#eventProcess">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="generatedTime">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="triggeredAccount">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="controlledRole">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="generatedAccount">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="generatedArtifact">...</owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#eventTime">...</owl:ObjectProperty>
  <owl:DatatypeProperty rdf:ID="noLater">...</owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="name">...</owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="noEarlier">...</owl:DatatypeProperty>
</rdf:RDF>

```

Figura 4.32: TBox do modelo OPM em OWL-DL


```

<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:a="http://www.ipaw.info/2007/opm#" xmlns:b="http://openprovenance.org/property/" xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <a:Account rdf:about="id:_null"></a:Account>
  <a:Used rdf:about="tag:tupeloproject.org,2006:086bc990c6572936f31f74c1fbb548d067633594">
    <a:eventAccount rdf:resource="id:orange"/>
    <a:usedArtifact rdf:resource="id:a4"/>
    <a:usedByProcess rdf:resource="id:p3"/>
    <a:usedRole>
      <a:Role rdf:about="tag:tupeloproject.org,2006:36504f0acc48ce3f8e5fac15a8512277072832b2">
        <rdfs:label>used</rdfs:label>
      </a:Role>
    </a:usedRole>
    <rdfs:label>used</rdfs:label>
  </a:Used>
  <a:Used rdf:about="tag:tupeloproject.org,2006:18008357e6f4579b760ed3bbf0e697f5aeb72caa"></a:Used>
  <a:Used rdf:about="tag:tupeloproject.org,2006:1a464225d22654f12caffa1c0d73d0e31cbb04a8"></a:Used>
  <a:Used rdf:about="tag:tupeloproject.org,2006:28e7ec2c632da69321fb26350d0584e848d50929"></a:Used>
  <a:Derived rdf:about="tag:tupeloproject.org,2006:32302d246cd85e9061126bc23818789e3df705de"></a:Derived>
  <a:Derived rdf:about="tag:tupeloproject.org,2006:36890c14adb167fb8253b3e7c62d9e5603c67dae"></a:Derived>
  <a:Generated rdf:about="tag:tupeloproject.org,2006:41b5b404b682758ed296c019f2d2b23ce7e9f8ce"></a:Generated>
  <a:Generated rdf:about="tag:tupeloproject.org,2006:4281c0ba7fa3fde916ba7fc854674ccb8c2bb355"></a:Generated>
  <a:Controlled rdf:about="tag:tupeloproject.org,2006:58ac31d053612850ac0af71d70bcc662be6c41e9"></a:Controlled>
  <a:Derived rdf:about="tag:tupeloproject.org,2006:64ba3e4bd012d91b319d1145e8d90633455cf4ec"></a:Derived>
  <a:Used rdf:about="tag:tupeloproject.org,2006:6fbfb8d1e66335f6f4fec4f2c0349506efaf58f"></a:Used>
  <a:Used rdf:about="tag:tupeloproject.org,2006:7cc32a2a5b7dc7d6a993305569fc0e8d0f3eab2c"></a:Used>
  <a:Derived rdf:about="tag:tupeloproject.org,2006:7f7685b5246c64ec1417a29b9149b4fb134a3ca8"></a:Derived>
  <a:Controlled rdf:about="tag:tupeloproject.org,2006:8044effb37e84c17fbc58bf221518309e909bafd"></a:Controlled>
  <a:Generated rdf:about="tag:tupeloproject.org,2006:82ecd41d8a7c42cb49fafae36b8033d179ced97"></a:Generated>
  <a:Generated rdf:about="tag:tupeloproject.org,2006:83ec25c3be6b0bea141b583ae8ebbe3b80307f27"></a:Generated>
  <a:Derived rdf:about="tag:tupeloproject.org,2006:89d0ba8908f5dd1e7fad18033c3f400707e6f856"></a:Derived>
  <a:Generated rdf:about="tag:tupeloproject.org,2006:9e2bfcafd0b04503abd3a0a9d04987c5ea0c96fc"></a:Generated>
  <a:Derived rdf:about="tag:tupeloproject.org,2006:9f619d1eb583e0772c80a70d10cdc8135d8a7410"></a:Derived>
  <a:Generated rdf:about="tag:tupeloproject.org,2006:bf720447700a53449666e8323398dc6f33075db3"></a:Generated>
  <a:Derived rdf:about="tag:tupeloproject.org,2006:db7615b63e58ac6f6f47e9229ea5cbcafbb003ff"></a:Derived>
</rdf:RDF>

```

Figura 4.33: Grafo de proveniência OPM serializado em RDF/XML

Concluindo, a camada de geração do grafo de proveniência do *SciProv* utiliza a infraestrutura promovida pelo projeto *Open Provenance Model* — Tupelo, Sesame, *TBox* OPM — para construir uma representação em sintaxe RDF/XML serializada a partir do grafo de proveniência implementado em memória. A Figura 4.33 apresenta à saída em RDF/XML correspondente ao grafo de proveniência OPM cuja representação visual em PDF é exibida na Figura 4.31.

Cabe observar que a arquitetura do *SciProv* oferece o recurso de exibição do grafo de proveniência OPM serializado em RDF/XML a partir da interface com o usuário.

4.2.8 Implementação da Consulta ao Grafo de Proveniência

Conforme apresentado na Seção 4.2.7, o projeto *Open Provenance Model* promove a utilização da infraestrutura disponível no *middleware* Tupelo para a serialização do grafo de proveniência. Coerente com esta diretriz do modelo OPM, a arquitetura do *SciProv* emprega tal mecanismo de vinculação.

Entretanto, o *SciProv* não utiliza a infraestrutura do arcabouço Sesame para prover uma camada de consulta ao grafo de proveniência em SPARQL. O Sesame — até a versão

2.3 — não dá suporte a inferências a partir de OWL-DL e, portanto, não oferece recursos para a composição de consultas com base em uma semântica rica e complexa além do que é possível com RDF e mesmo com OWL-*Lite*.

Além disso, Sesame utiliza como linguagem de consulta padrão a *Sesame RDF Query Language* (SeRQL), que embora seja muito similar a SPARQL, possui sintaxe diversa.

Nesse cenário, o desenvolvimento do presente trabalho emprega o arcabouço Jena para prover o suporte a consultas na linguagem padrão SPARQL. Jena consiste em um projeto de código livre que oferece um ambiente baseado em APIs Java para a construção de aplicações *web* semânticas com suporte a RDF, RDFs, OWL e a linguagem de consulta SPARQL [63].

Jena incorpora também uma máquina de inferência, porém com poder de expressividade limitado, capaz de processar apenas até OWL *Lite*. Entretanto, Jena permite a composição de consultas semânticas a partir de máquinas de inferências externas que podem ser referenciadas pelo arcabouço.

Considerando-se que a proposta para o desenvolvimento da arquitetura do *SciProv* norteia-se pelo emprego de tecnologias *web* semânticas relevantes e atuais, a arquitetura implementada utiliza a máquina de inferência do arcabouço Pellet [64], que provê suporte para perfis até em OWL 2 — incluindo OWL 2 EL na versão 2.2.2.

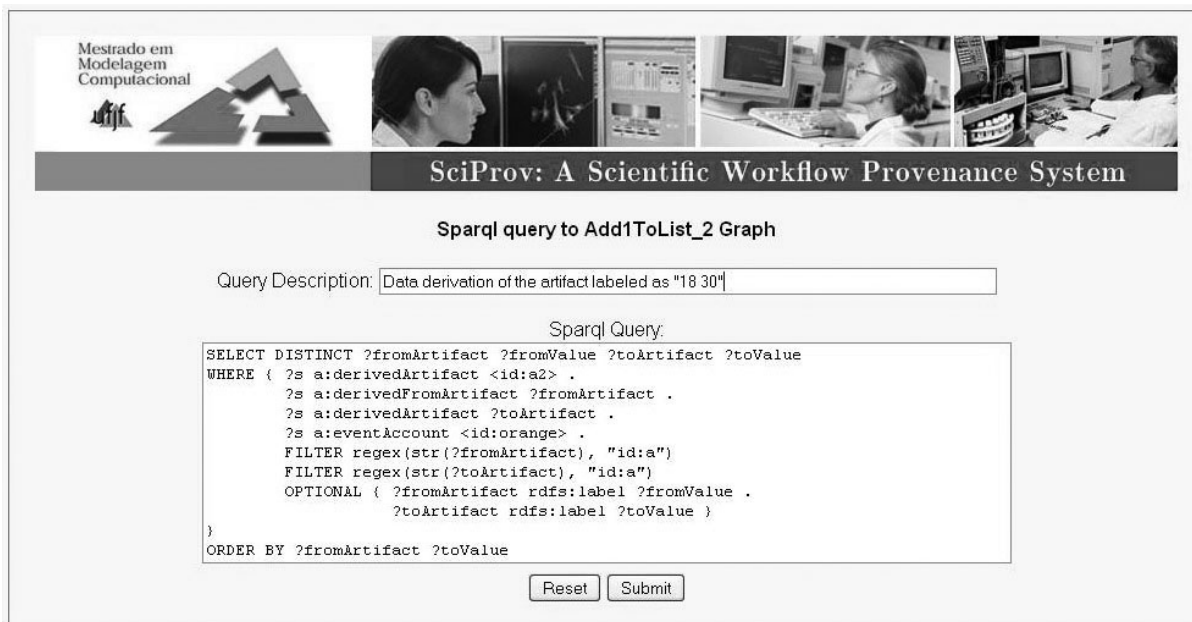
A possibilidade de processar inferências desde OWL-DL até OWL 2 EL através do *reasoner* Pellet confere ao *SciProv* um alto poder de expressividade em consultas SPARQL ao grafo de proveniência modelado a partir de tecnologias *web* semântica.

4.2.9 Consulta Semântica no SciProv

No contexto da arquitetura do *SciProv*, a consulta SPARQL apresentada na Figura 4.34 pode apresentar resultados distintos dependendo tanto da especificação da ontologia que modela o conhecimento sobre o domínio estudado quanto da ontologia que representa o modelo OPM. Além disso, o resultado também é sensível ao poder de expressividade da máquina de inferência utilizada para o processamento de consultas aos metadados, por exemplo, uma máquina de inferência capaz de processar ontologias em OWL-DL é capaz de processar consultas semânticas mais expressivas do que outra em OWL *Lite*.

A Figura 4.34 exibe uma consulta SPARQL ao grafo de proveniência cuja representação visual foi apresentada anteriormente na Figura 4.31. No contexto desse exemplo e

considerando-se a ontologia do modelo OPM conforme proposto pelo projeto *Open Provenance Model*, o resultado da consulta semântica exibido a partir da interface com o usuário do *SciProv* é mostrada na Figura 4.35.



Mestrado em Modelagem Computacional

SciProv: A Scientific Workflow Provenance System

Sparql query to Add1ToList_2 Graph

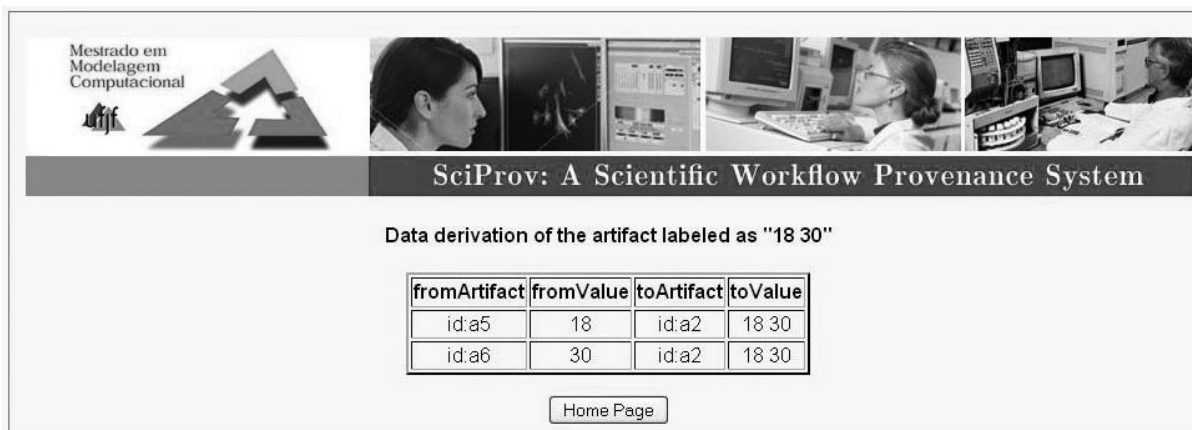
Query Description:

Sparql Query:

```
SELECT DISTINCT ?fromArtifact ?fromValue ?toArtifact ?toValue
WHERE {
  ?s a:derivedArtifact <id:a2> .
  ?s a:derivedFromArtifact ?fromArtifact .
  ?s a:derivedArtifact ?toArtifact .
  ?s a:eventAccount <id:orange> .
  FILTER regex(str(?fromArtifact), "id:a")
  FILTER regex(str(?toArtifact), "id:a")
  OPTIONAL {
    ?fromArtifact rdfs:label ?fromValue .
    ?toArtifact rdfs:label ?toValue }
}
ORDER BY ?fromArtifact ?toValue
```

Figura 4.34: Interface com o usuário do *SciProv* para consulta em SPARQL

Pode-se observar que esse resultado está coerente com o grafo de proveniência OPM da Figura 4.31, onde o nó representado pelo artefato cujo rótulo é “18 29” possui arestas do tipo *wasDerivedFrom* com os nós rotulados como “18” e “30”. Cabe observar que esta relação de dependência causal deve ser explicitamente declarada no processo de instrumentalização do *workflow* científico modelado uma vez que configura uma regra de completude inválida segundo a ontologia do padrão OPM (vide 4.2.2).



Mestrado em Modelagem Computacional

SciProv: A Scientific Workflow Provenance System

Data derivation of the artifact labeled as "18 30"

fromArtifact	fromValue	toArtifact	toValue
id:a5	18	id:a2	18 30
id:a6	30	id:a2	18 30

Figura 4.35: Resultado da consulta SPARQL ao grafo de proveniência em RDF

4.2.10 Consulta Semântica com Perfil Adaptado do modelo OPM

O OPM prevê a possibilidade de construção de especializações a serem incorporadas ao modelo na forma de perfis, conforme abordado na Seção 4.1.11. Esta característica do projeto OPM constitui um dos aspectos mais importantes contemplados na arquitetura do *SciProv*. O emprego de um perfil especializado tem por objetivo obter resultados mais abrangentes em buscas semânticas aos dados de proveniência coletados no domínio de *workflows* científicos.

No exemplo de *workflow* científico modelado conforme apresentado e instrumentalizado para a coleta dos metadados de proveniência na Seção 4.2.1 (ver Figura 4.19) é empregada uma especialização ao modelo OPM no contexto de *workflows* científicos que define relacionamentos adicionais à ontologia proposta pelo projeto *Open Provenance Model*. O Apêndice A contém a ontologia do modelo OPM especializada para o domínio de *workflows* científicos definida no presente trabalho.

A Figura 4.36 apresenta um aspecto de um perfil OPM no domínio de *workflows* científicos. Conforme discutido previamente na Seção 4.1.11, que declara as propriedades denominadas *derivedArtifact* e *derivedFromArtifact* como sendo propriedades inversas e transitivas. Isso implica dizer que em uma abordagem de proveniência em um contexto de *workflows* científicos, estas propriedades configuradas como inversas e transitivas estabelecem um relacionamento de dependência causal do tipo *was derived from* ao longo do processo de transformação de um dado desde a sua origem até o resultado final.

O conteúdo semântico especializado conforme apresentado na Figura 4.36 permite que novas relações de dependência causal do tipo *wasDerivedFrom* possam ser inferidas a partir de máquinas de inferência com capacidade de processamento de ontologias em OWL-DL com suporte a propriedades inversas do tipo objeto (*object properties*) — como o Pellet, utilizado na implementação da arquitetura do *SciProv*.

O resultado apresentado na Figura 4.37 corrobora os objetivos propostos ao se incorporar à camada de perfis do modelo OPM uma especialização no domínio de *workflows* científicos. Como pode ser observado, a arquitetura do *SciProv* é capaz de processar o conteúdo semântico acrescentado ao modelo e como resultado, apresentar um grafo de proveniência OPM que exhibe tanto as relações de dependência causal do tipo *wasDerivedFrom* declaradas explicitamente a partir do *workflow* instrumentalizado (Figura 4.25)

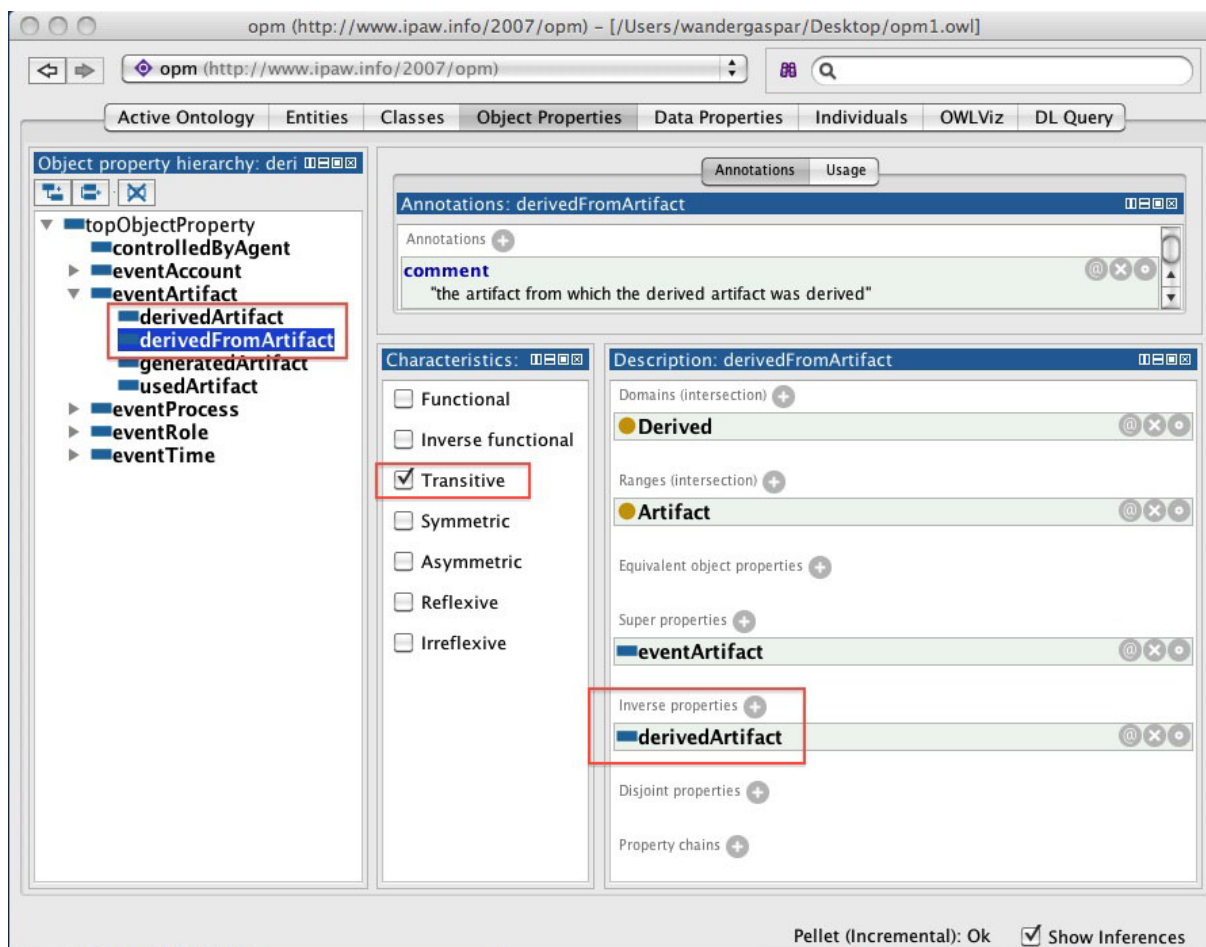


Figura 4.36: Aspecto da especialização da ontologia do modelo OPM

quanto inferidas a partir da especialização do modelo OPM proposta no presente trabalho (Figura 4.36).

Neste novo cenário, a partir de um perfil especializado para o modelo OPM no domínio de *workflows* científicos, a arquitetura do *SciProv* é capaz de processar as inferências do tipo *wasDerivedFrom* e apresentar o resultado da consulta SPARQL conforme exibido anteriormente na Figura 4.34 com todas as possibilidades visualizadas na Figura 4.36. A interface com o usuário do *SciProv* com os resultados inferidos pode ser visto na Figura 4.38.

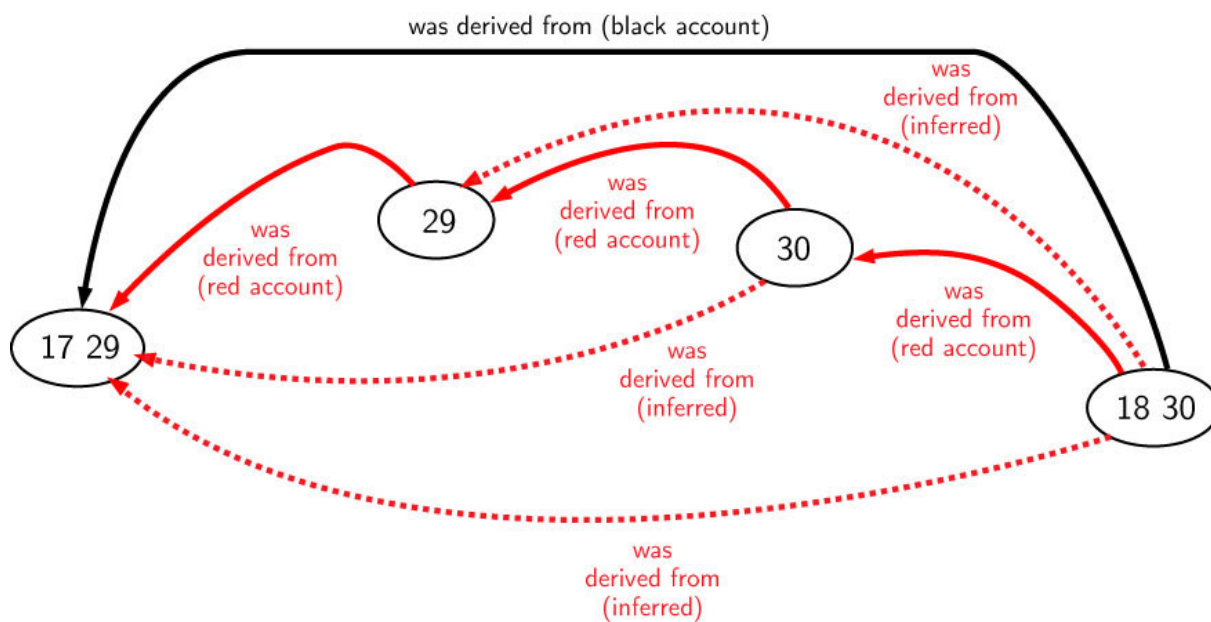




Figura 4.37: Relações de Dependência Inferidas no Grafo de Proveniência

Mestrado em Modelagem Computacional





SciProv: A Scientific Workflow Provenance System

Data derivation of the artifact labeled as "18 30"

fromArtifact	fromValue	toArtifact	toValue
id:a1	17 29	id:a3	17
id:a1	17 29	id:a5	18
id:a1	17 29	id:a2	18 30
id:a1	17 29	id:a4	29
id:a1	17 29	id:a6	30
id:a3	17	id:a5	18
id:a3	17	id:a2	18 30
id:a4	29	id:a2	18 30
id:a4	29	id:a6	30
id:a5	18	id:a2	18 30
id:a6	30	id:a2	18 30

[Home Page](#)

Figura 4.38: Interface com o usuário do *SciProv* com resultados inferidos

5 ESTUDOS DE CASO

Este Capítulo apresenta um conjunto de estudos de caso cujo objetivo principal consiste em avaliar a adequabilidade da arquitetura proposta à coleta e gerência da proveniência de dados e processos no contexto de experimentos científicos processados através de simulações em ambientes de pesquisa colaborativa, dispersos geograficamente e interconectados através de uma grade computacional. Além disso, o conjunto de estudos de caso são utilizados no sentido de avaliar o desempenho da execução dos *workflows* científicos instrumentalizados para a captura da proveniência segundo o modelo OPM.

O processo de concepção, execução e avaliação do conjunto de estudos de casos propostos neste trabalho procurou seguir os princípios descritos por Kitchenhan *et al.* [65], segundo os quais um bom estudo de caso pode prover informação suficiente para ajudar a julgar se uma metodologia específica poderá trazer benefícios em um determinado contexto.

Ainda segundo Kitchenhan *et al.* [65], três etapas distintas devem constar na elaboração de um estudo de caso:

- Planejamento, que consiste na definição do estudo de caso, identificando-se os objetivos a serem alcançados, de acordo com a situação que se deseja experimentar.
- Monitoramento, que refere-se ao acompanhamento da execução das etapas do estudo de caso, registrando-se o tempo de duração, as dificuldades encontradas e os resultados obtidos.
- Avaliação dos resultados, que consiste na descrição dos resultados gerais obtidos no estudo de caso, com base nos dados obtidos na etapa de monitoramento.

O conjunto de estudos de casos discutidos neste Capítulo foram executados em um computador configurado com processador AMD Turion 64 X2 Mobile Technology TL-50 1.60 GHz cache 2×256kB, memória RAM 3 GB DDR2 533/667 MHz, chipset AMD M690, memória de vídeo compartilhada, disco rígido de 80 GB SATA 5400 rpm e sistema operacional Microsoft Windows XP Professional versão 2002 com service pack 3.

Já considerando-se a etapa de planejamento [65], foram empregados para a consecução dos objetivos propostos neste Capítulo, os recursos disponíveis na infraestrutura computacional e nas bases de dados do Banco de Dados de DNA do Japão (DDBJ). O trabalho

do DDBJ consiste em prover e gerir um repositório de dados e processos à comunidade científica mundial na área de Bioinformática.

5.1 Banco de Dados de DNA do Japão

O Banco de Dados de DNA do Japão (*DNA Data Bank of Japan*)¹ encontra-se localizado no Instituto Nacional de Genética (NIG – *National Institute of Genetics*) daquele país e está incluído como membro do consórcio internacional de pesquisa científica colaborativa na área de sequenciamento genético de nucleotídeos denominado (INSDC – *International Nucleotide Sequence Database Collaboration*) [66].

Além disso, ainda em um contexto de pesquisa científica colaborativa em um ambiente distribuído, o DDBJ promove o intercâmbio dos dados disponíveis com o Laboratório de Biologia Molecular Europeu (EMBL – *European Molecular Biology Laboratory*) e com o *GenBank* norte-americano, localizado no Centro Nacional de Informação em Biotecnologia (*National Center for Biotechnology Information*).

O DDBJ possui um comitê consultivo internacional composto por nove membros entre cientistas japoneses, europeus e norte-americanos. Há também um comitê de colaboração que presta consultoria em diversas questões técnicas relacionadas com a cooperação científica em nível internacional.

Juntamente com o EMBL e o *GenBank*, a função do DDBJ consiste em coletar as informações de sequências nucleotídicas obtidas experimentalmente e manter um repositório de dados interoperável construído de acordo com o padrão adotado pelas entidades envolvidas. As informações incluem referências de natureza biológica sobre os dados coletados das pesquisas desenvolvidas, como a origem de organismos estudados, função do gene e outras propriedades do sequenciamento genético.

Especificamente, os estudos de caso implementados no presente trabalho utilizam a interface denominada *Web API for Biology* (WABI)² do DDBJ que reúne especificações de serviços *web* SOAP e REST, tutoriais, ferramentas de pesquisa e modelos de *workflows* construídos a partir da infraestrutura disponibilizada.

¹DDBJ disponível em <http://www.ddbj.nig.ac.jp/index-e.html>

²WABI disponível em <http://xml.nig.ac.jp/index.html>

5.1.1 Serviço *GetEntry* do DDBJ

Para efeito de composição de *workflows* científicos necessários à configuração de estudos de casos optou-se por um serviço *web* denominado *GetEntry*, que permite configurar uma busca abrangente na base de dados DDBJ e retornar informações específicas de acordo com os parâmetros de pesquisa formulados. *GetEntry* possui 45 métodos distintos (em dezembro/2010) que permitem a composição de buscas ao repositório DDBJ.

Vale mencionar que não cabe formular no contexto do presente trabalho uma justificativa com base em requisitos de um domínio específico da Ciência para a escolha de uma base de dados e de um repositório de serviços *web* em particular. A arquitetura do *SciProv* tem por objetivo prover um mecanismo de coleta e gerência dos dados de proveniência independente do domínio do experimento científico, do conjunto de dados originalmente acessados e gerados e dos serviços *web* invocados durante o processo de execução dos *workflows* científicos.

Nesse contexto, deve-se mencionar que a escolha da infraestrutura computacional disponibilizada pelo Banco de Dados de DNA do Japão para a elaboração dos estudos de casos apresentados relaciona-se às características associadas à pesquisa colaborativa, processamento de *workflows* em ambiente remoto e distribuído e à invocação de serviços *web*.

O método *getDDBJEntry* do serviço *GetEntry* retorna um amplo conjunto de informações relacionadas ao sequenciamento de um organismo específico, incluindo-se especificações técnicas do campo da Genética além de dados adicionais que descrevem o trabalho de pesquisa desenvolvido, dados dos cientistas envolvidos e publicações relacionadas.

Ainda na etapa de planejamento, [65] a Figura 5.1 exibe uma representação abstrata do *workflow* científico utilizado para a orquestração dos estudos de caso. As seções seguintes detalham a implementação desse modelo a partir dos SGWfCs Kepler e Taverna em casos de uso que procuram avaliar a adequabilidade do *SciProv* no que concerne à coleta e ao tratamento dos metadados de proveniência no contexto de experimentos científicos processados através de simulações em ambientes de pesquisa colaborativa dispersos geograficamente e interconectados através de uma grade computacional.

Conforme apresentado na Figura 5.1, inicialmente são definidos os parâmetros referentes ao serviço *web* a ser executado (URL do DDBJ, serviço *web* a ser invocado, método utilizado e código de acesso a um determinado organismo). Com base nos parâmetros

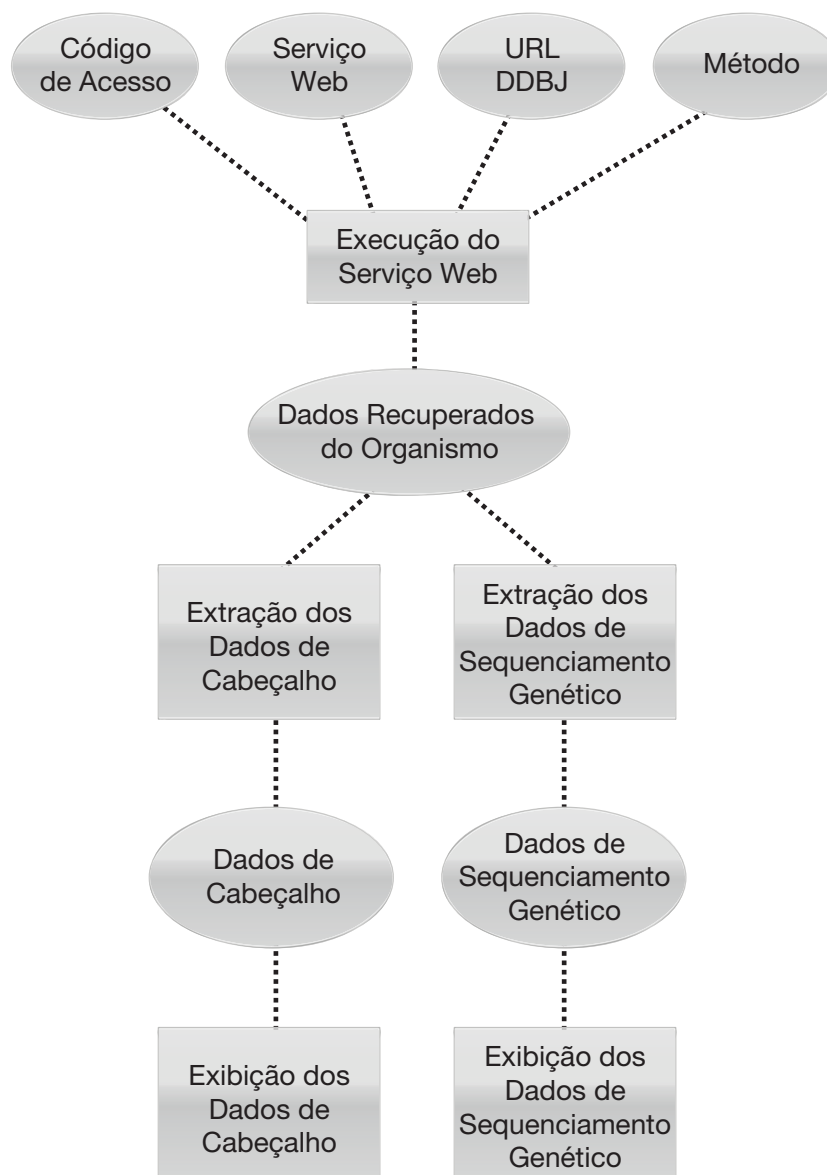


Figura 5.1: Representação abstrata do *workflow* científico

informados, o serviço *web* *GetEntry* do DDBJ é executado remotamente e retornado um conjunto de dados sobre o organismo especificado. Em uma etapa seguinte, os dados obtidos são processados localmente, ou seja, no mesmo nó computacional onde o *workflow* científico é instanciado, com o objetivo de extrair-se dois grupos de informações. O primeiro grupo extraído deve conter o registro de cabeçalho do organismo recuperado, incluindo-se a versão do sequenciamento genético disponível na base de dados do DDBJ, o nome científico e a classificação biológica. O segundo grupo deve conter o detalhamento do sequenciamento genético. Por fim, após a extração das informações requeridas, os dados obtidos são exibidos ao usuário.

5.1.2 Estudo de Caso com o SGWfC Kepler

O primeiro estudo de caso especificado emprega o SGWfC Kepler para compor um *workflow* científico simplificado que acessa a base de dados do DDBJ através da invocação do serviço *web GetEntry*. As informações pesquisadas referem-se a um determinado organismo cujos dados do sequenciamento genético encontram-se disponíveis para busca. Em seguida, o *workflow* deve ainda extrair dos dados retornados do DDBJ dois conjuntos distintos de informações que foram denominados *Header* e *Gene Sequence*. Para efeito de ilustração, a Figura 5.2 e a Figura 5.3 apresentam, respectivamente, os dados de saída a serem gerados pelo *workflow* modelado no Kepler para *Header* e *Gene Sequence* referentes ao organismo *Synechococcus elongatus* PCC 7942.

```

LOCUS      AB000100                2992 bp    DNA      linear    BCT 15-MAY-2009
DEFINITION Synechococcus elongatus PCC 7942 genes for intrinsic membrane
           protein, malK-like protein, cyanase, complete cds.
ACCESSION  AB000100
VERSION    AB000100.1
KEYWORDS   .
SOURCE     Synechococcus elongatus PCC 7942
  ORGANISM Synechococcus elongatus PCC 7942
           Bacteria; Cyanobacteria; Chroococcales; Synechococcus.

```

Figura 5.2: Informações de cabeçalho de *Synechococcus elongatus*

Como observação, o sequenciamento genético do organismo *Synechococcus elongatus*, conforme retornado pelo serviço *web GetEntry*, é constituído de 2992 genes. A Figura 5.3 apresenta apenas a parte inicial dos dados processados.

A Figura 5.4 apresenta o *workflow* científico modelado no SGWfC Kepler para busca, recuperação e tratamento dos dados de sequenciamento genético do organismo *Synechococcus elongatus* a partir dos dados disponíveis no DDBJ.

O componente *DDBJ Service* encapsula uma invocação ao serviço *web GetEntry* e passa como parâmetro de entrada o código de acesso AB000100, referente ao organismo *Synechococcus elongatus*. Ainda na Figura 5.4, vale observar que os componentes compostos denominados *Extract Reader* e *Extract Gene Sequence* são responsáveis, respectivamente, pela extração das informações apresentadas anteriormente nas Figuras 5.2 e 5.3.

O resultado do processo de instrumentalização do *workflow* científico modelado no Kepler é apresentado na Figura 5.5. O objetivo da instrumentalização implementada

ORIGIN

```

1 ctgcagccgc cgactgaaat ctatcgggaa gaaaagctcg cttacgacac ctttaaccgg
61 caggatccag tcgcttacct cgcactctca aagcagaaat acgggagata aacacaactt
121 atggtgagaa ctccgtgacc gctttaccta cgttggggcg tctccatcct cagcgtgctt
181 gcgttcctag ccatttggca aattgcccga gcttcaggat ttttaggcaa aacttttctt
241 ggctccctgc gcactttgca ggatttggtt ggatggcttt cagatccctt ctttgataac
301 ggccccaatg acttagggat tggctggaac ttactgatta gtttgcgctg cgttgcgatc
361 ggctacctgc tggcaacagt tgttgcaatt ctttggggga ttgcaatcgg tatgtcggcg
421 ctagettcca gtatttttcc gccctttgtg caactcctga agccagtttc accttgggcc
481 tggttgcccga ttgggtctctt cttattccga gattcgggat tgacgggtgt ttttgtcate

```

Figura 5.3: Sequenciamento genético de *Synechococcus elongatus* (parte inicial)

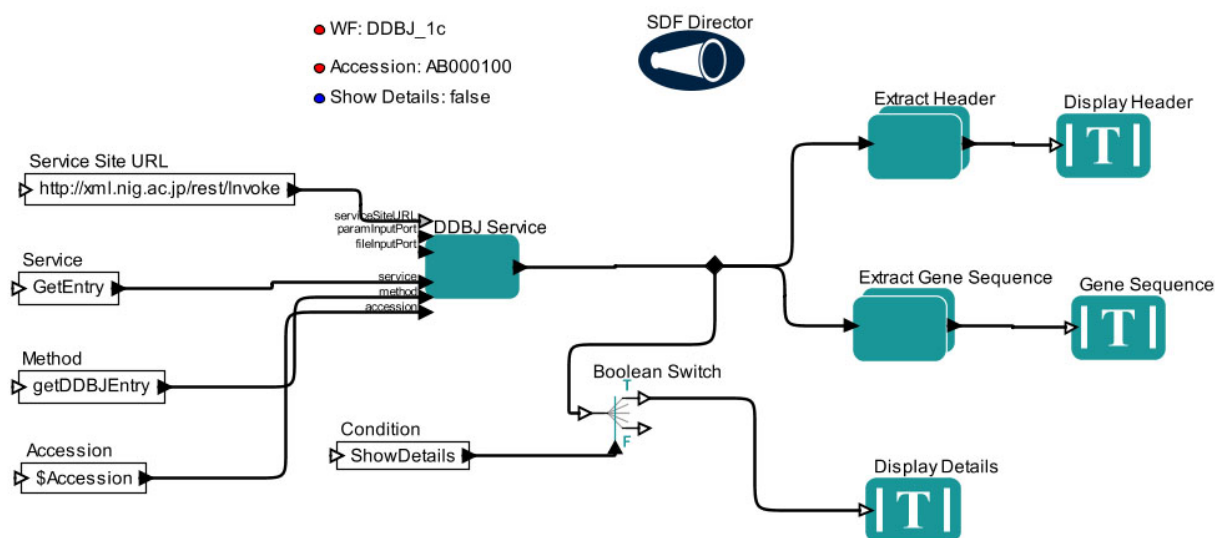


Figura 5.4: *Workflow* para sequenciamento genético no Kepler

consiste em capturar explicitamente todas as entidades e dependências de causalidade observadas no *workflow* segundo o modelo abstrato OPM.

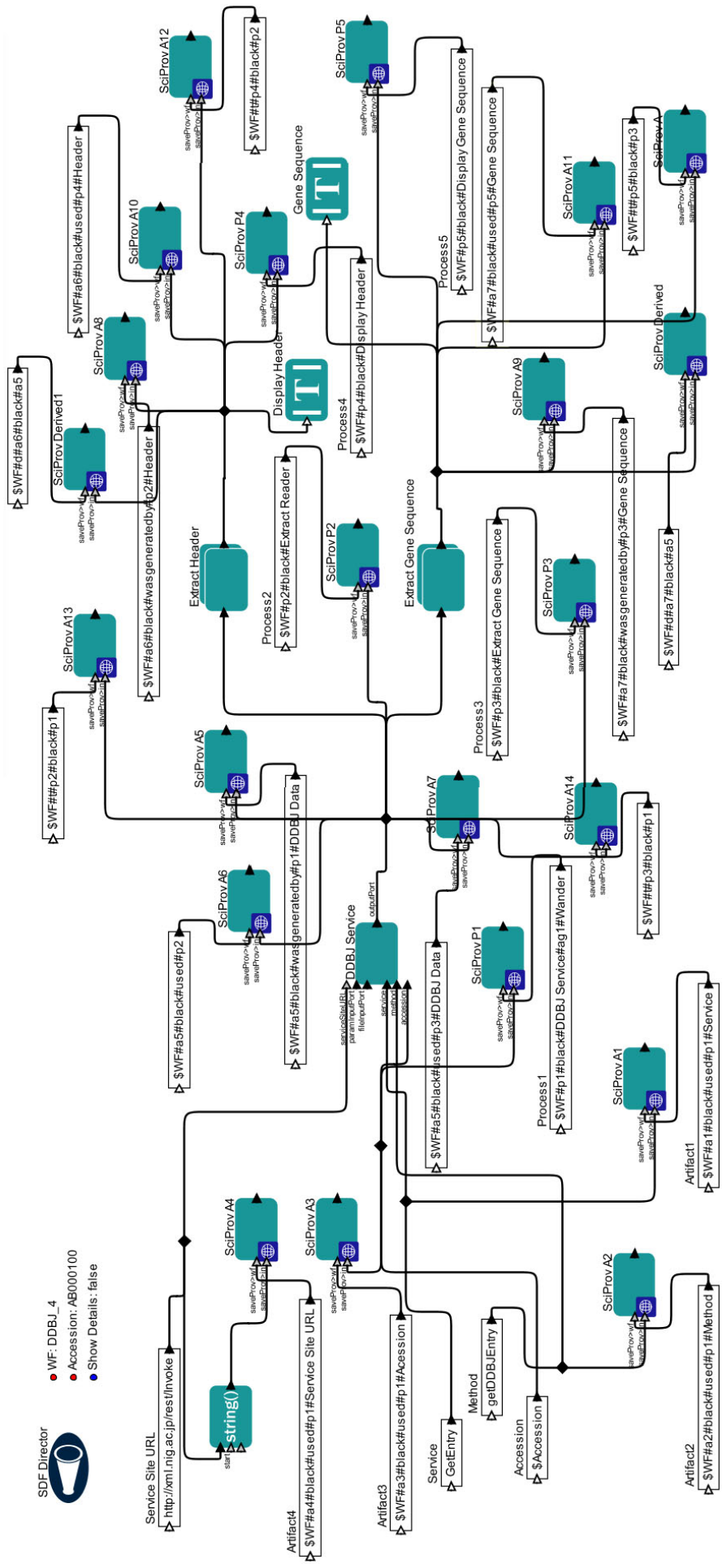


Figura 5.5: Workflow instrumentalizado para sequenciamento genético no Kepler

Na etapa de monitoramento [65], uma vez instrumentalizado o *workflow* científico modelado no Kepler para busca e extração dos dados do sequenciamento genético de um organismo armazenados no DDBJ, o *SciProv* encontra-se habilitado a coletar e armazenar os metadados de proveniência gerados ao longo do processo de execução. Conforme apresentado na Seção 4.1.6, os metadados capturados segundo o modelo abstrato OPM foram inicialmente persistidos em uma base de dados relacional.

Particularmente nesse estudo de caso, após a execução do *workflow* científico para a busca e extração dos dados de sequenciamento genético referentes ao organismo *Synechococcus elongatus*, seguida da coleta e armazenamento dos metadados de proveniência pelo *SciProv*, foi possível a construção de uma infraestrutura baseada em tecnologia *web* semântica para representação e consulta às informações de linhagem.

Nesse contexto, conforme detalhamento apresentado na Seção 4.1.8, a arquitetura do *SciProv* promoveu a vinculação dos metadados de proveniência em formato XML (Figura 5.6) e em formato RDF/XML (Figura 5.7).



```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opm:opmGraph xmlns:opm="http://openprovenance.org/model/v1.1.a"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://example.com/">
  <opm:accounts>
    <opm:account id="orange"/>
    <opm:account id="black"/>
    <opm:overlaps>...</opm:overlaps>
  </opm:accounts>
  <opm:processes>
    <opm:process id="p1">...</opm:process>
    <opm:process id="p3">...</opm:process>
    <opm:process id="p2">...</opm:process>
    <opm:process id="p5">...</opm:process>
    <opm:process id="p4">...</opm:process>
  </opm:processes>
  <opm:artifacts>
    <opm:artifact id="a4">...</opm:artifact>
    <opm:artifact id="a3">...</opm:artifact>
    <opm:artifact id="a2">...</opm:artifact>
    <opm:artifact id="a1">...</opm:artifact>
    <opm:artifact id="a5">...</opm:artifact>
    <opm:artifact id="a7">...</opm:artifact>
    <opm:artifact id="a6">...</opm:artifact>
  </opm:artifacts>
  <opm:agents>...</opm:agents>
  <opm:causalDependencies>...</opm:causalDependencies>
</opm:opmGraph>

```

Figura 5.6: Metadados de proveniência gerados em sintaxe XML

Além do processo de vinculação em formatos XML e RDF/XML, o *SciProv* oferece um recurso, conforme apresentado na Seção 4.1.8, capaz de promover a visualização das entidades (artefatos, processos e agentes) e relações de causalidade observadas durante

```

<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:a="http://www.ipaw.info/2007/opm#"
  xmlns:b="http://openprovenance.org/property/"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <a:Account rdf:about="id:_null"></a:Account>
  <a:Account rdf:about="id:orange"></a:Account>
  <a:Used rdf:about="tag:tupeloproject.org,2006:104a8f0acd1c85256efa653aded36365dace0f8e"></a:Used>
  <a:Used rdf:about="tag:tupeloproject.org,2006:3c6b0d420b02ad76e8fdcf2c58a85e42f9ea8408"></a:Used>
  <a:Derived rdf:about="tag:tupeloproject.org,2006:4ec54dfa2800bffb56700a745314c57a16ed562b"></a:Derived>
  <a:Used rdf:about="tag:tupeloproject.org,2006:5aeea725c0a1abda62e19672e1bbd82d8e03d48"></a:Used>
  <a:Generated rdf:about="tag:tupeloproject.org,2006:5ede263efe534a7246fe54e8ab8602a97fa76e8b"></a:Generated>
  <a:Used rdf:about="tag:tupeloproject.org,2006:955ca1f227581a6fd4d8f3bf38e60257a843eca4"></a:Used>
  <a:Used rdf:about="tag:tupeloproject.org,2006:97ec8e166b5073ebc6a85615b22ebf541fafe8db"></a:Used>
  <a:Used rdf:about="tag:tupeloproject.org,2006:a3d46eaa7ffe072f715b67910e43e7b750aa4d46"></a:Used>
  <a:Triggered rdf:about="tag:tupeloproject.org,2006:ae7225ff8b6281e653cad62e3b8427c318049678"></a:Triggered>
  <a:Triggered rdf:about="tag:tupeloproject.org,2006:bbd684b4d416abbc6d6fa137c7f3b8321aeab2f2"></a:Triggered>
  <a:Triggered rdf:about="tag:tupeloproject.org,2006:c06d8b8afe8a57dd3baef0c9d538992e4ed40407"></a:Triggered>
  <a:Triggered rdf:about="tag:tupeloproject.org,2006:c6a40e214ef9c569f9521db538adad044b908d6e"></a:Triggered>
  <a:Used rdf:about="tag:tupeloproject.org,2006:cbc481515a7b42ab1874cc438c93937811c97793"></a:Used>
  <a:Generated rdf:about="tag:tupeloproject.org,2006:d2a4f44c95964c1a9d7e59eeaf4d5a4dab16c2bf"></a:Generated>
  <a:Controlled rdf:about="tag:tupeloproject.org,2006:d69a459c95251662a6e7a0ee0395e46266a01769"></a:Controlled>
  <a:Derived rdf:about="tag:tupeloproject.org,2006:f2ca1877a6736ea74acc8dca53c53b02a901d220"></a:Derived>
  <a:Used rdf:about="tag:tupeloproject.org,2006:f5c69495f75252ac0363218538666dc33315bfcbb"></a:Used>
  <a:Generated rdf:about="tag:tupeloproject.org,2006:fe209171f1dc6d275fe80e27b5490ec6dd37f82eb"></a:Generated>
</rdf:RDF>

```

Figura 5.7: Metadados de proveniência gerados em sintaxe RDF/XML

o processo de execução do *workflow* científico a partir de uma representação gráfica do grafo de proveniência (Figura 5.8).

Além de demonstrar a aplicabilidade do *SciProv* em um cenário coerente com a arquitetura proposta, um dos objetivos deste estudo de caso foi formular uma busca semântica ao grafo de proveniência gerado. A consulta procurou extrair informações que não estivessem explicitamente representadas e que, portanto, foram inferidas com base no conhecimento sobre o modelo OPM descrito no componente *TBox* da ontologia especializada para o domínio de *workflows* científicos (ver Seção 4.1.11).

Uma dos refinamentos incluídos na especialização ao OPM proposto no presente trabalho está relacionado às propriedades (*object properties*) denominadas *triggeredProcess* e *triggeredByProcess* da ontologia OWL que descreve o modelo. Tais propriedades representam relacionamentos entre as classes *Triggered* e *Process* e, de acordo com a ontologia original do OPM, não são consideradas inversas e transitivas.

A propriedade *triggeredProcess* liga indivíduos da classe *Triggered* (domínio) a indivíduos da classe *Process* (alcance ou *range*). Esta propriedade relaciona as entidades OPM do tipo Processo que foram geradas a partir de uma relação de dependência causal do tipo *was triggered by*. Conforme apresentado na Figura 5.9, o modelo especializado proposto pela arquitetura do *SciProv* declara tanto a característica inversa com a propriedade *triggeredByProcess* quanto transitiva para a propriedade *triggeredProcess*.

A partir de uma pesquisa semântica formulada através da linguagem SPARQL (Figura

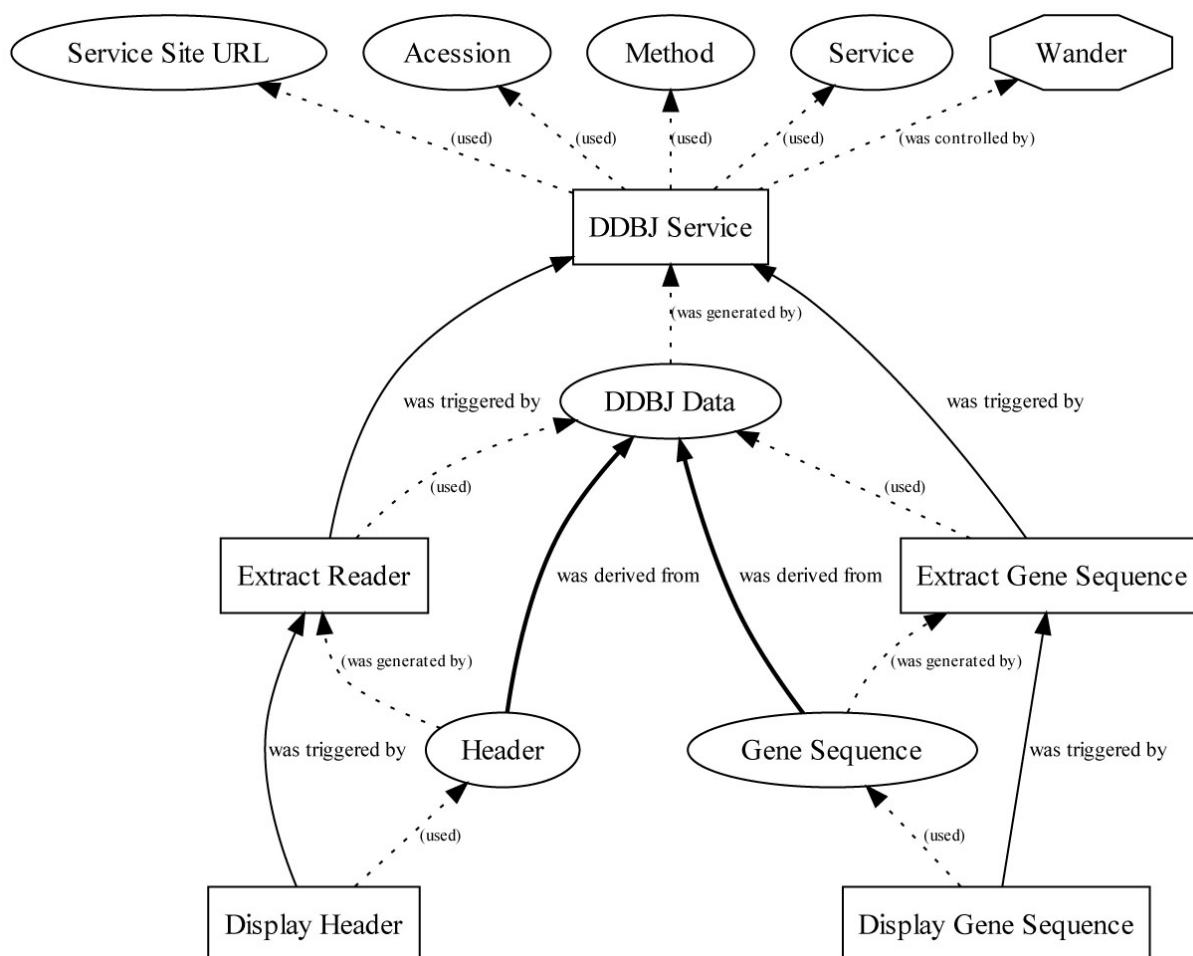


Figura 5.8: Grafo de proveniência do *workflow* científico modelado no Kepler

5.10), o *SciProv* foi capaz de retornar informações inferidas com base no conhecimento sobre o modelo OPM especializado e que não se encontram representadas de forma explícita nos metadados de proveniência.

Na etapa de avaliação dos resultados [65], pode-se observar na Figura 5.11 que as duas primeiras relações de causalidade (em destaque) não foram coletadas diretamente a partir do mecanismo de instrumentalização inserido no *workflow* científico apresentado no presente estudo de caso (ver Figura 5.8). Portanto, tais resultados devem-se aos recursos de inferência disponíveis no *SciProv* e obtidos a partir das características inversa e transitiva incluídas nas propriedades *triggeredProcess* e *triggeredByProcess* do *TBox* que descreve o modelo especializado OPM no domínio de *workflows* científicos.

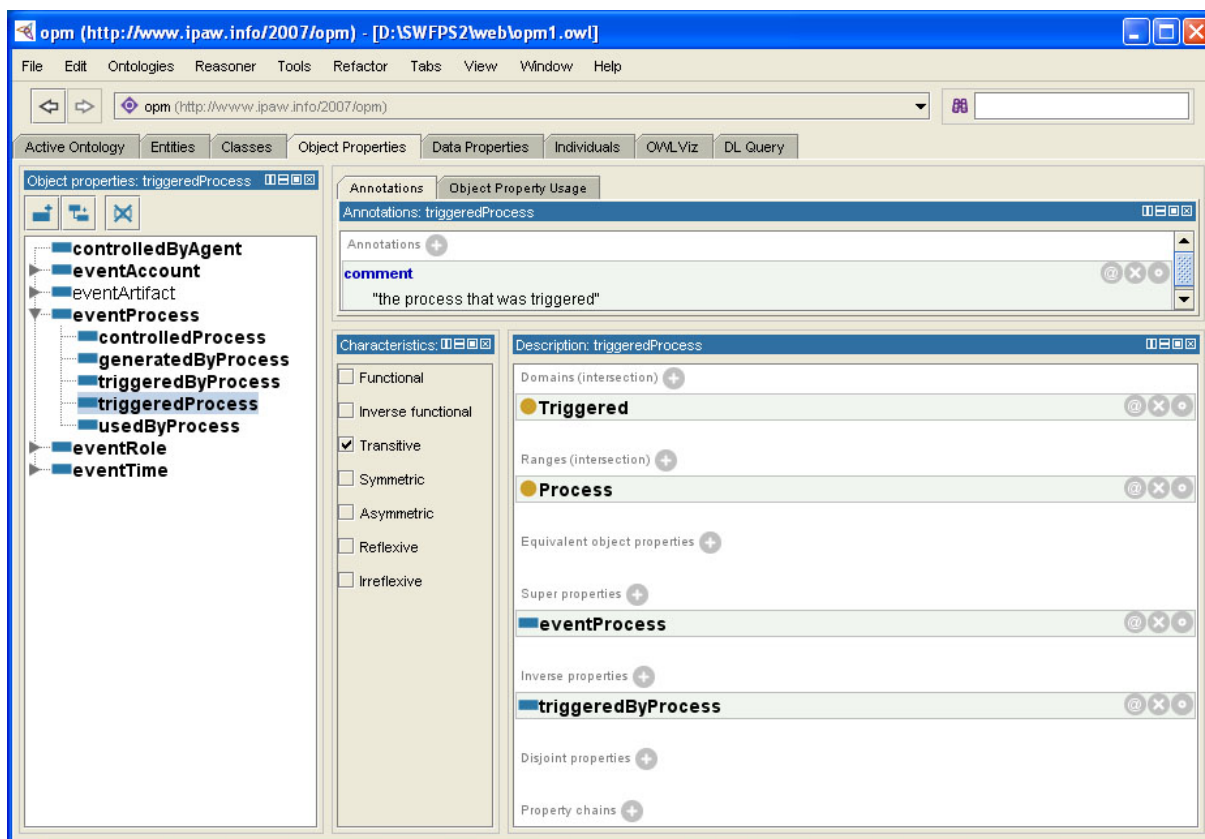


Figura 5.9: Características inversa e transitiva para a propriedade *triggeredProcess*

Mestrado em Modelagem Computacional

SciProv: A Scientific Workflow Provenance System

Sparql query to DDBJ_4 Graph

Query Description:

Sparql Query:

```

SELECT DISTINCT ?fromProcess ?fromValue ?toProcess ?toValue
WHERE {
  ?s a:triggeredByProcess <id:p1> .
  ?s a:triggeredProcess ?toProcess .
  ?s a:triggeredByProcess ?fromProcess .
  ?s a:eventAccount <id:black> .
  FILTER regex(str(?fromProcess), "id:p")
  FILTER regex(str(?toProcess), "id:p")
  OPTIONAL { ?fromProcess rdfs:label ?fromValue .
             ?toProcess rdfs:label ?toValue }
}
ORDER BY ?fromProcess ?toValue

```

Figura 5.10: Pesquisa semântica em SPARQL no *workflow* científico modelado no Kepler

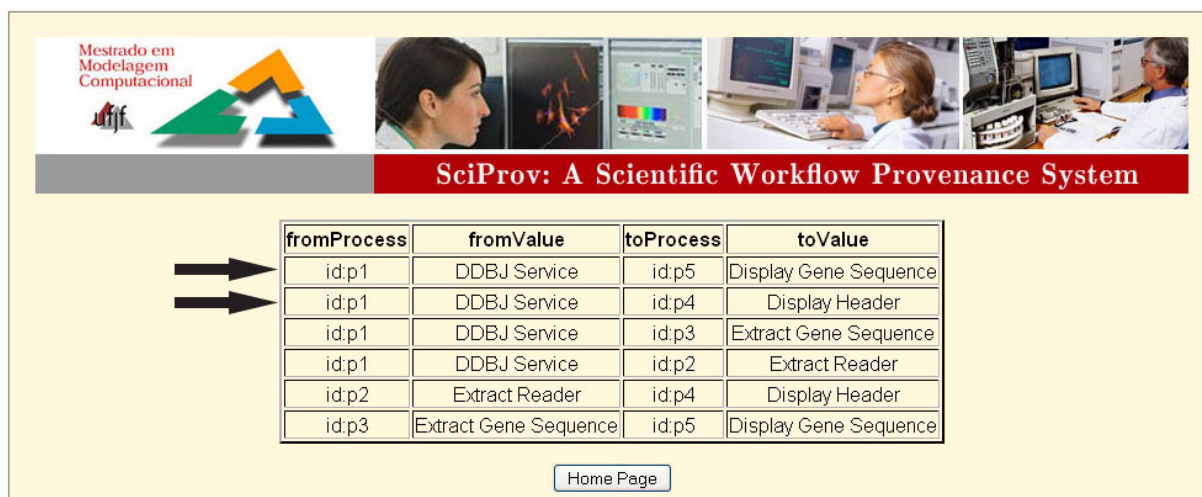


Figura 5.11: Resultado da busca semântica no *workflow* científico com inferências em destaque

5.1.3 Estudo de Caso com o SGWfC Taverna

O segundo estudo de caso construído emprega o SGWfC Taverna para compor um *workflow* científico cuja funcionalidade deve ser idêntica ao do *workflow* construído na seção anterior onde foi utilizado o SGWfC Kepler. O objetivo desse caso de uso é demonstrar que a arquitetura proposta do *SciProv* é capaz de gerar metadados e grafos de proveniência equivalentes a partir de SGWfC distintos.

Nesse contexto, embora Kepler e Taverna possuam interfaces e metodologias diversas para a construção de *workflows*, deseja-se avaliar que experimentos científicos funcionalmente equivalentes modelados em SGWfCs diversos podem ser instrumentalizados pelo *SciProv* de forma a coletar e processar os metadados de proveniência de forma uniforme de acordo com o modelo OPM.

Vale acrescentar que o objetivo do presente estudo de caso é importante no que se refere ao escopo pretendido para o *SciProv*, no sentido de prover uma infraestrutura para a coleta e gerência de metadados de proveniência de forma independente do SGWfC utilizado para a orquestração do experimento científico.

Assim, considerando-se a etapa de planejamento, a Figura 5.12 exibe um *workflow* científico modelado no SGWfC Taverna para acessar a base de dados do DDBJ através da invocação do serviço *web GetEntry*. O *workflow* deve ainda extrair dos dados retornados do DDBJ dois conjuntos distintos de informações denominados *Header* e *Gene Sequence*.

Para efeito de ilustração, as Figuras 5.13 e 5.14 apresentam, respectivamente, os dados

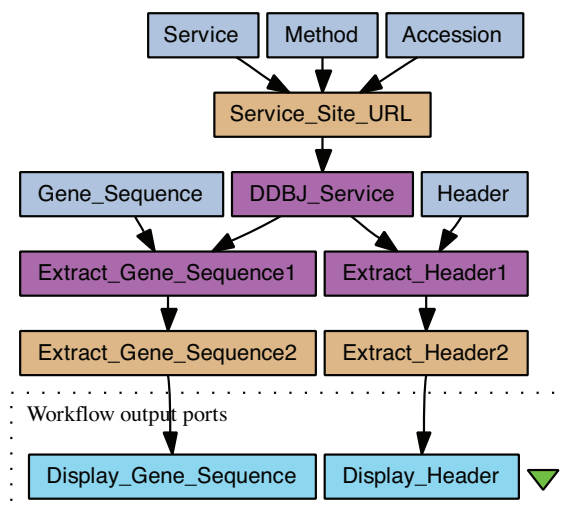


Figura 5.12: *Workflow* científico modelado no SGWfC Taverna

de saída a serem gerados no Taverna para *Header* e *Gene Sequence* referentes ao organismo *Synechococcus elongatus* PCC 7942.

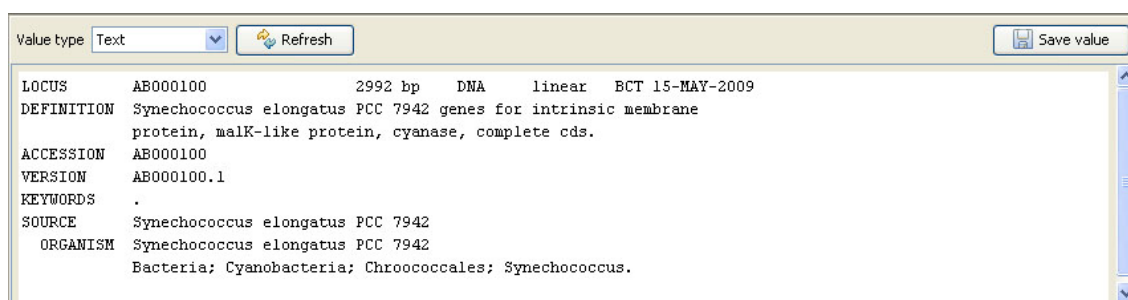


Figura 5.13: Informações de cabeçalho de *Synechococcus elongatus* no Taverna

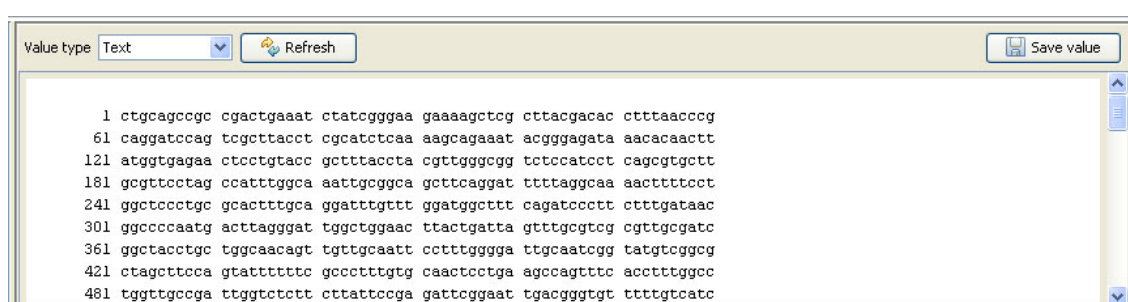


Figura 5.14: Sequenciamento genético de *Synechococcus elongatus* no Taverna

O processo de instrumentalização do *workflow* científico modelado no Taverna é apresentado na Figura 5.15. A instrumentalização deve capturar explicitamente todas as entidades e dependências de causalidade observadas no *workflow* segundo o modelo abstrato OPM e, além disso, deve ser consistente com o processo análogo realizado para o *workflow* científico modelado anteriormente no SGWfC Kepler.

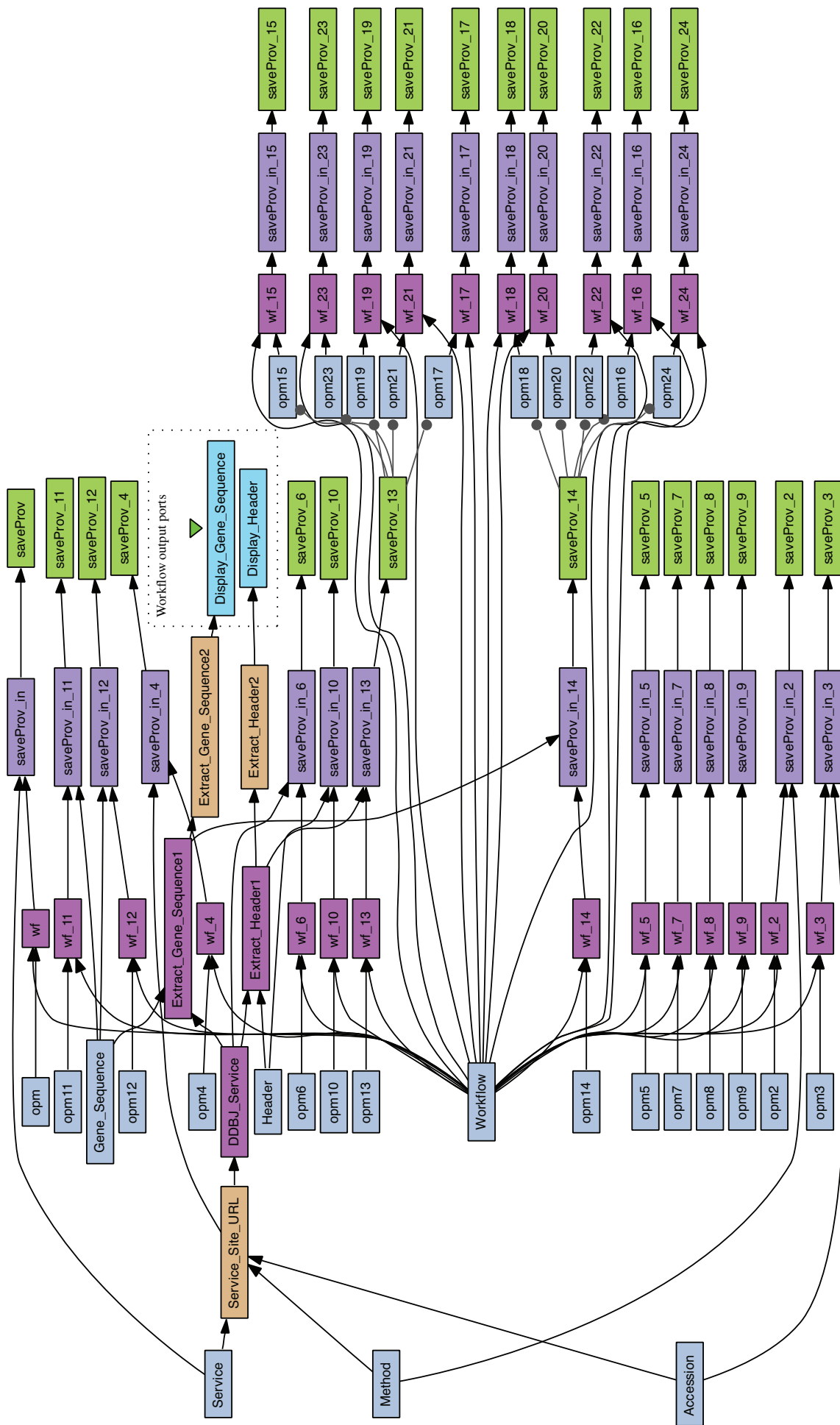


Figura 5.15: *Workflow* instrumentalizado para sequenciamento genético no Taverna

Na etapa de monitoramento, após a instrumentalização do *workflow* científico modelado no Taverna para o tratamento do sequenciamento genético a partir dos dados disponíveis no DDBJ, o *SciProv* pode coletar e armazenar os metadados de proveniência gerados durante a execução. Inicialmente os metadados capturados segundo o modelo abstrato OPM são persistidos em um banco de dados relacional.

Em uma etapa seguinte à persistência dos metadados de proveniência, a arquitetura do *SciProv* promove a vinculação nos formatos XML e RDF/XML. Também é gerada a representação gráfica do grafo de proveniência cujo objetivo é exibir as entidades OPM (artefatos, processos e agentes) e as relações de causalidade observadas durante o processo de execução do *workflow* científico (Figura 5.16).

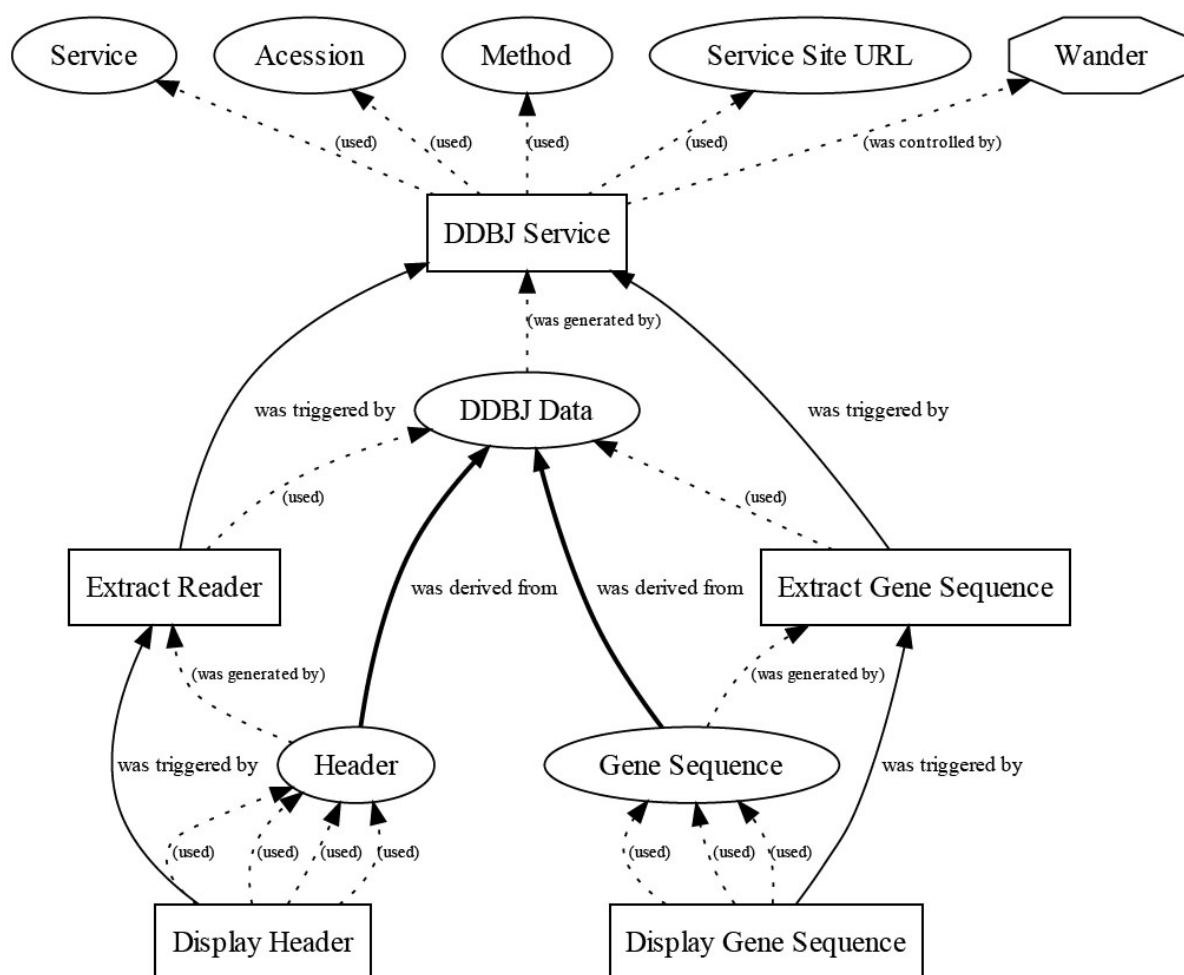
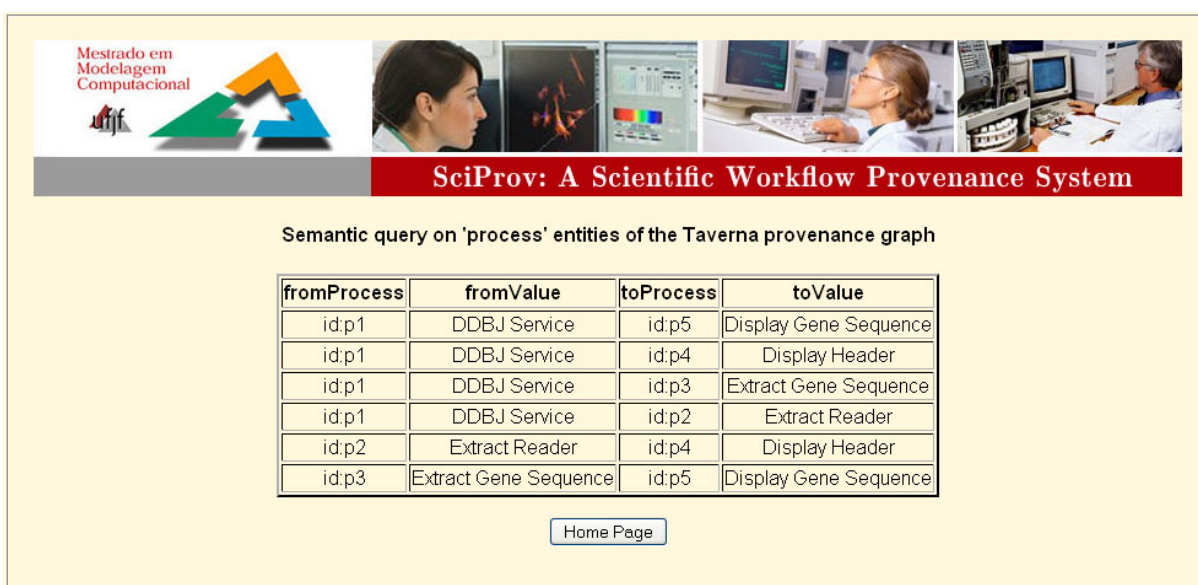


Figura 5.16: Grafo de proveniência do *workflow* científico modelado no Taverna

Na etapa de avaliação, observa-se através das Figuras 5.8 e 5.16 que ocorreu uma equivalência entre os grafos de proveniência gerados como resultado da execução dos *workflows* científicos modelados para a execução em SGWfCs distintos (Kepler e Taverna).

Considerando-se que ambos os *workflows* foram projetados para o processamento com funcionalidades idênticas, resguardando-se as particularidades de cada *software*, o resultado mostra-se coerente com a proposta da arquitetura do *SciProv*, no sentido de gerar metadados de proveniência de acordo com o modelo abstrato OPM de forma independente do SGWfC empregado.

A Figura 5.17 apresenta o resultado da pesquisa semântica aos metadados de proveniência gerados a partir da execução do *workflow* científico modelado no SGWfC Taverna. De forma coerente com o esperado, são apresentados valores idênticos àqueles obtidos utilizando-se o grafo de proveniência produzido pela execução do *workflow* de igual funcionalidade construído no SGWfC Kepler (ver Figura 5.11).



The screenshot displays the SciProv interface. At the top, there is a banner with the text "Mestrado em Modelagem Computacional" and "SciProv: A Scientific Workflow Provenance System". Below the banner, a navigation bar contains the text "Semantic query on 'process' entities of the Taverna provenance graph". The main content area features a table with the following data:

fromProcess	fromValue	toProcess	toValue
id:p1	DDBJ Service	id:p5	Display Gene Sequence
id:p1	DDBJ Service	id:p4	Display Header
id:p1	DDBJ Service	id:p3	Extract Gene Sequence
id:p1	DDBJ Service	id:p2	Extract Reader
id:p2	Extract Reader	id:p4	Display Header
id:p3	Extract Gene Sequence	id:p5	Display Gene Sequence

At the bottom of the interface, there is a button labeled "Home Page".

Figura 5.17: Resultado da busca semântica no *workflow* científico modelado no Taverna

5.1.4 Avaliação de Desempenho de Workflows Instrumentalizados

Conforme apresentado na Seção 1.2, um dos aspectos a serem avaliados no presente trabalho consiste em medir o impacto da coleta e persistência dos metadados de proveniência sobre o desempenho dos experimentos científicos instrumentalizados pelo *SciProv*.

Nesse contexto, os estudos de caso apresentados nas Seções 5.1.2 e 5.1.3 permitem elaborar alguns estudos comparativos simples referentes aos tempos de execução de *workflows* científicos adaptados para a captura e armazenamento dos metadados gerados.

Na etapa de planejamento, para a determinação dos tempos de execução, os *workflows* modelados nos SGWfC Kepler e Taverna foram modificados para registro do instante de início e de término do processamento, com precisão de milésimos de segundo. Na etapa de monitoramento, os *workflows* foram executados no ambiente descrito no início do Capítulo e os resultados registrados para posterior avaliação.

A Tabela 5.1 apresenta os resultados obtidos para três medições do *workflow* científico orquestrado no Kepler, tanto para a versão original (Figura 5.4) quanto para a versão instrumentalizada para coleta e persistência dos metadados de linhagem (Figura 5.5). Computou-se os valores absolutos das diferenças de tempo de execução, apresentados com precisão de milésimos de segundo e os valores relativos, que indicam o acréscimo percentual de tempo em relação ao original, para a execução do *workflow* científico instrumentalizado pelo *SciProv*.

Tabela 5.1: Tempos de execução para o *workflow* modelado no Kepler

Execução	Sem Instr.	Com Instr.	Absoluto	Relativo
1	1s875	2s953	1s078	57
2	1s922	2s750	0s828	44
3	1s922	4s438	2s516	131
Média	1s906	3s380	1s474	77

Descontando-se o processamento referente a invocação ao serviço *web* remoto *GetEntry*, obtém-se tempos de execução para o *workflow* científico modelado no Kepler conforme apresentado na Tabela 5.2.

Tabela 5.2: Tempos de execução para o *workflow* modelado no Kepler descontando-se o processamento do serviço *web* *GetEntry*

Execução	Sem Instr.	Com Instr.	Absoluto	Relativo
1	0s109	0s422	0s313	387
2	0s063	0s406	0s343	644
3	0s063	0s407	0s344	646
Média	0s078	0s412	0s334	528

As medições apresentadas nas Tabelas 5.1 e 5.2 referem-se a execuções unitárias do *workflow* modelado no SGWfC Kepler. Para avaliar o desempenho a partir de um conjunto maior de execuções, tanto a versão original quanto a versão instrumentalizada do *workflow*

sofreram adaptações para repetir o processamento a cada execução. A Tabela 5.3 exibe os resultados obtidos para 1, 10 e 100 iterações do *workflow* modelado no Kepler.

Tabela 5.3: Tempos de execução para 1, 10 e 100 iterações

Execução	Sem Instr.	Com Instr.	Absoluto	Relativo
1	1s906	3s380	1s474	77
10	18s578	27s125	8s547	46
100	3min10s818	4min11s500	1min0s682	32

Pelos valores medidos, observa-se uma gradual diminuição na degradação de performance do *workflow* instrumentalizado em relação ao *workflow* original modelado no Kepler.

A Tabela 5.4 apresenta o resultado das medições realizadas a partir do *workflow* modelado no SGWfC Taverna, tanto para a versão original quanto para a versão instrumentalizada pelo *SciProv* para a coleta e persistência dos metadados de proveniência.

Tabela 5.4: Tempos de execução para o *workflow* modelado no Taverna

Execução	Sem Instr.	Com Instr.	Absoluto	Relativo
1	5s610	8s140	2s530	45
2	2s500	3s515	1s015	41
3	2s547	3s047	1s196	20
Média	3s552	4s901	1s380	38

Considerando-se a etapa de avaliação dos resultados, a partir das medições de tempo registradas é possível fazer algumas considerações:

- Considerando-se os valores das Tabelas 5.1 e 5.4, observa-se que o acréscimo de tempo médio (em valores absolutos) para uma execução do *workflow* científico instrumentalizado em relação ao original é aproximadamente igual para ambos os SGWfCs.
- Considerando-se os valores da Tabelas 5.1 e e 5.2, observa-se um significativo aumento no tempo relativo de execução quando se desconsidera o serviço *web* remoto.
- O acréscimo de tempo médio (em valores relativos) para uma execução do *workflow* científico instrumentalizado em relação ao original é menor no Taverna em compa-

ração com o Kepler. Isso deve ao tempo mais elevado de execução do *workflow* no SGWfC Taverna.

- Há um decréscimo nos tempos de execução do *workflow* instrumentalizado tanto no Kepler quanto no Taverna à medida que se amplia o número de iterações.

Observa-se que tais considerações devem ser avaliadas somente no contexto específico do *workflow* científico modelado no estudo de caso. É importante acrescentar que os tempos de execução de um *workflow* cuja orquestração inclui serviços *web* localizados remotamente dependem de uma série de fatores externos como a disponibilidade dos serviços, das taxas de transmissão de dados em rede e da infraestrutura computacional empregada. Portanto, tais aspectos em conjunto podem influenciar de forma relevante o desempenho de execução de *workflows* científicos executados em uma grade computacional [67].

Por outro lado, no caso de *workflows* científicos cujo processamento seja local, o impacto da instrumentalização no desempenho será tanto maior quanto menor for a complexidade do algoritmo referente ao experimento científico modelado computacionalmente.

Contudo, com base nas medições realizadas, considera-se importante que trabalhos futuros relacionados à arquitetura do *SciProv* tratem a questão do desempenho de uma forma específica uma vez que os valores relativos médios encontrados podem impor restrições à utilização do sistema em situações críticas de processamento. Nesse cenário, julga-se válido elaborar medições e avaliações estatísticas adequadas objetivando-se detectar gargalos no procedimento de coleta e armazenamento dos metadados de proveniência e propor refinamentos ao projeto.

5.1.5 Estudo de Caso em Conjunto com os SGWfCs Taverna e Kepler

O terceiro estudo de caso tem por objetivo apresentar uma situação que pode ser interessante em um contexto de experimentação científica em ambientes colaborativos. Especificamente, o que se vislumbra é a concepção de *workflows* modelados em conjunto utilizando-se dois ou mais SGWfCs. Assim, pode-se considerar que uma parte do trabalho seja desenvolvido em um determinado centro de pesquisa utilizando-se o Taverna como ferramenta para orquestração computacional do experimento e que outra parte do

workflow seja modelado em uma localidade distinta por um grupo de cientistas a partir do SGWfC Kepler.

O cenário acima descrito é avaliado nesse estudo de caso considerando-se ainda o acesso à base de dados do DDBJ através da invocação do serviço *web GetEntry*. O objetivo é capturar os metadados de proveniência gerados tanto durante a execução do *workflow* modelado no Taverna quanto no Kepler e produzir um grafo de linhagem único que seja consistente com os grafos obtidos nos casos de uso anteriores apresentados nesse trabalho.

Na etapa de planejamento, a parte inicial do *workflow* científico modelado em conjunto emprega o SGWfC Taverna, cujo resultado do processo de instrumentalização é exibido na Figura 5.18.

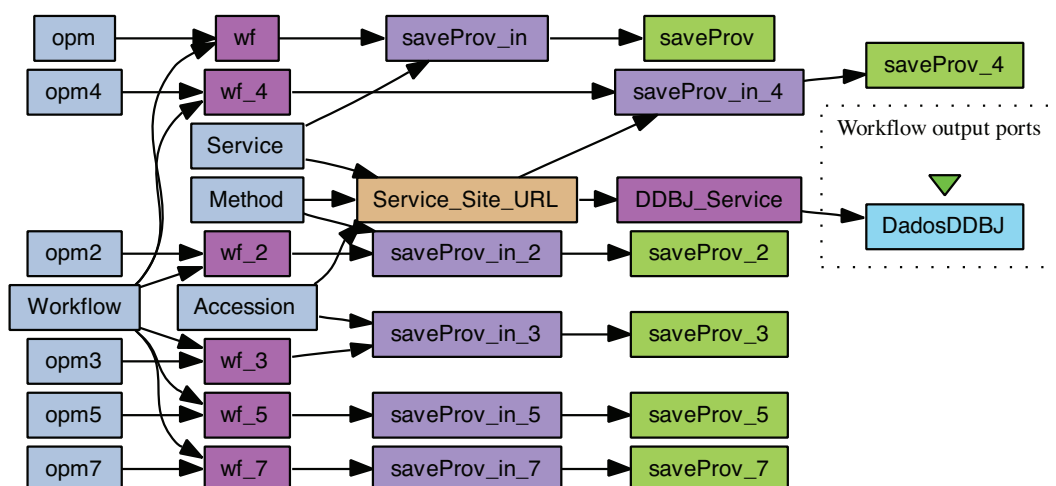


Figura 5.18: Parte do *workflow* científico conjunto modelado no Taverna

A parte final do *workflow* científico modelado em conjunto apresentado nesse estudo de caso emprega o SGWfC Kepler e o resultado do processo de instrumentalização é exibido na Figura 5.19. Deve-se observar que essa parte do *workflow* recebe como parâmetro de entrada os dados de saída produzidos a partir do Taverna (em destaque).

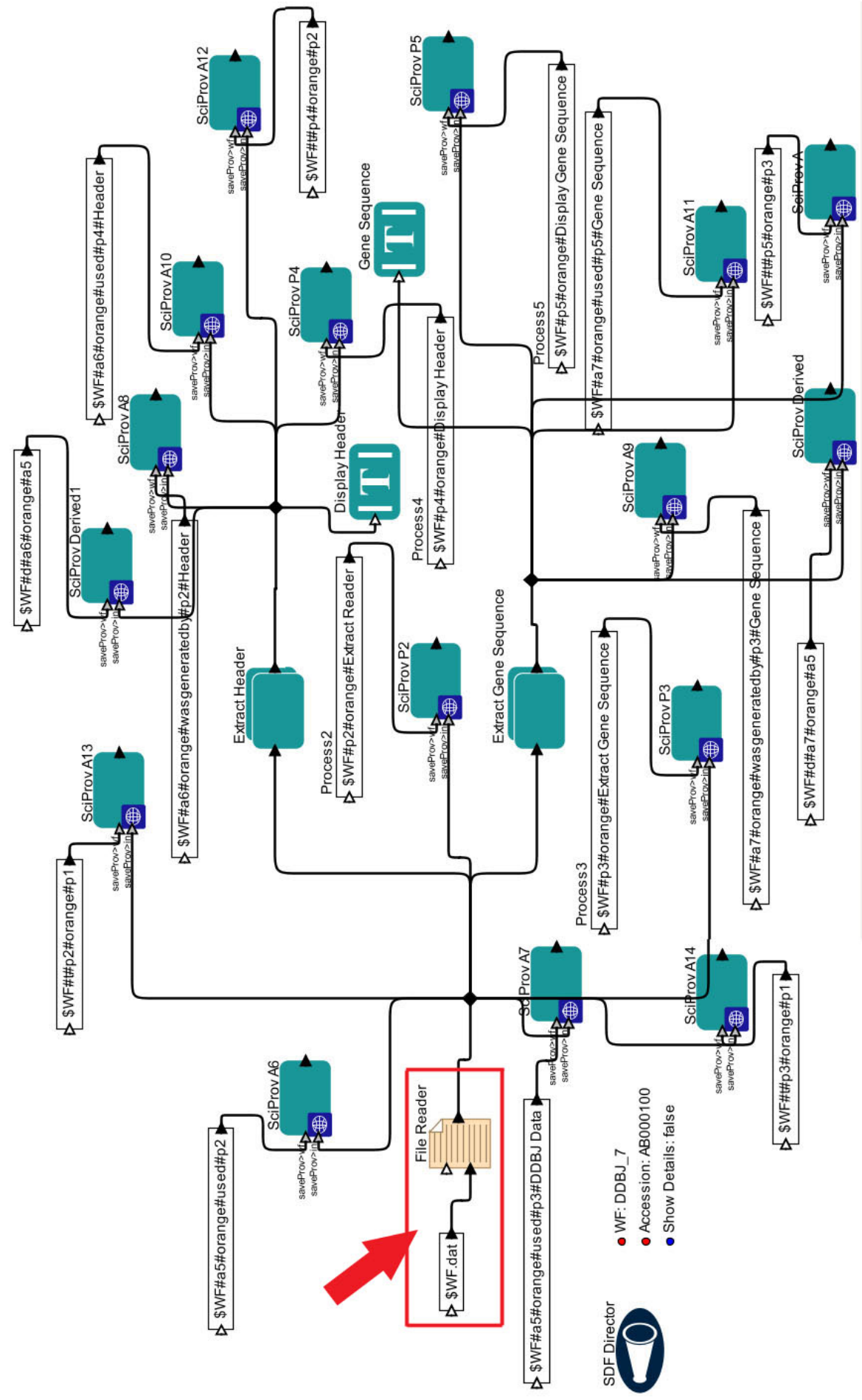


Figura 5.19: Parte do workflow científico modelado no Kepler

Ainda na etapa de planejamento, deve-se observar que tanto a execução do *workflow* científico modelado no Taverna quanto no Kepler devem receber a mesma identificação para que o processo de modelagem conjunta seja interpretado de forma correta pela arquitetura do *SciProv*. Esse cuidado permitirá que a captura dos metadados de proveniência relacionem-se a um mesmo grafo de proveniência no padrão OPM.

Uma vez que nesse estudo de caso são gerados metadados de proveniência a partir de SGWfC distintos, adotou-se o emprego das cores preto e vermelho para a identificação das descrições (*accounts*) OPM originadas a partir do Taverna e Kepler respectivamente. Essa abordagem tem por objetivo facilitar o entendimento quanto à origem dos metadados em um *workflow* científico modelado de forma conjunta.

Referente a etapa de monitoramento, a Figura 5.20 apresenta o grafo de proveniência gerado segundo o modelo abstrato OPM para uma execução do *workflow* científico modelado em conjunto no Taverna e Kepler. A partir da análise do resultado, pode-se observar que a linhagem dos metadados manteve-se coerente com aqueles obtidos nos estudos de caso apresentados anteriormente (Figuras 5.8 e 5.16). Esse resultado é interessante no sentido de adequar-se à proposta formulada para a arquitetura do *SciProv* de coletar e tratar os metadados de proveniência com base no modelo OPM em um ambiente colaborativo de pesquisa científica, disperso geograficamente e interligado a partir de recursos computacionais em grade.

Ainda na etapa de análise dos resultados, a Figura 5.21 apresenta o retorno da busca semântica aos metadados de linhagem gerados a partir da execução do *workflow* científico modelado em conjunto utilizando-se os SGWfCs Taverna e Kepler. Pode-se observar que os valores apresentados são consistentes com aqueles obtidos a partir dos grafos de proveniência gerados pela execução de *workflows* de igual funcionalidade construídos no SGWfC Kepler (ver Figura 5.11) e Taverna (ver Figura 5.17).

5.1.6 Estudo de caso com Arcabouço de Anotações OPM

Conforme apresentado na Seção 3.6, o modelo abstrato de proveniência OPM permite anexar informações adicionais na forma de anotações às entidades e dependências causais. O recurso é implementado pelo OPM através de um arcabouço específico, onde as anotações são tratadas na forma independente das entidades observadas no modelo [54].

No contexto da arquitetura do *SciProv*, o interesse pelo arcabouço reside na possibi-

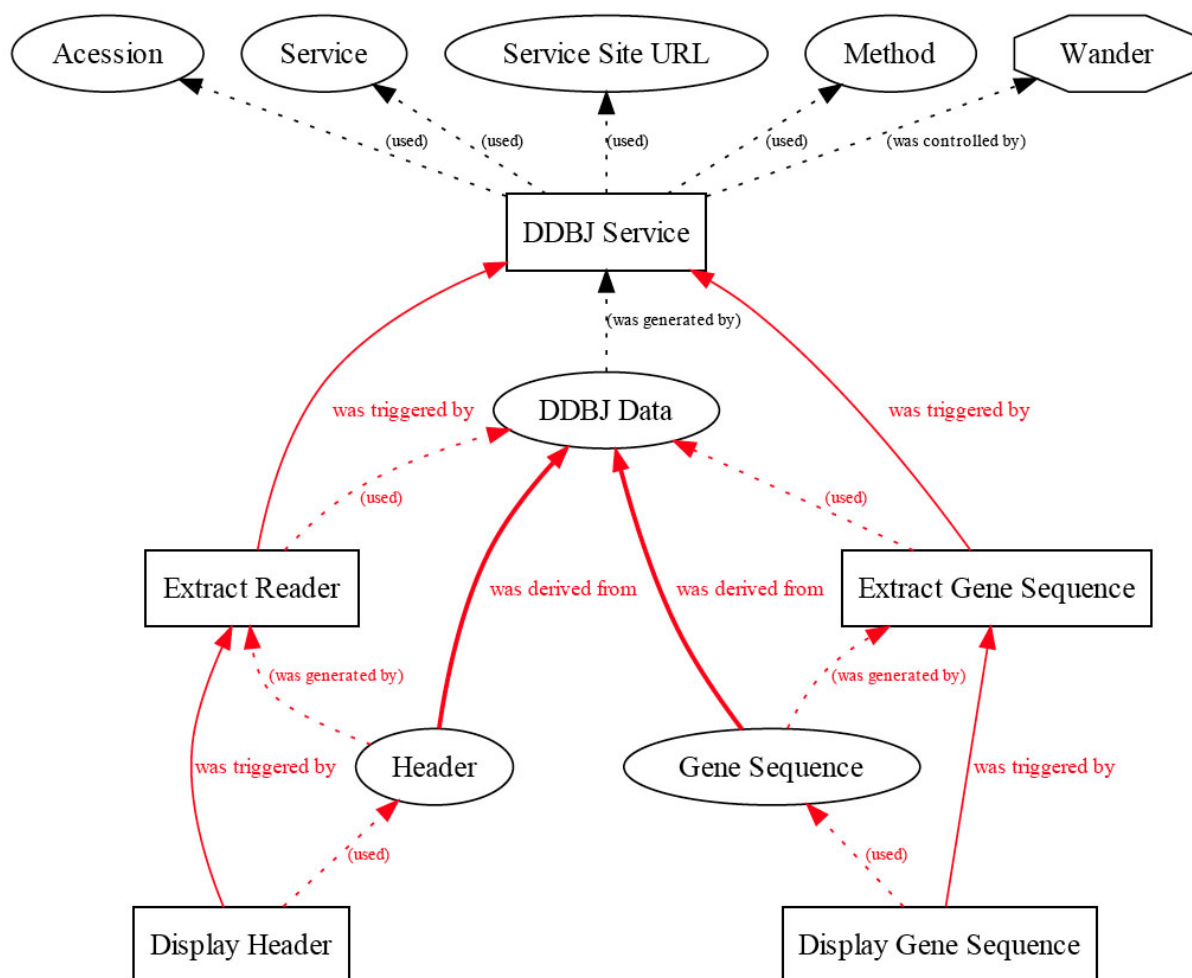




Figura 5.20: Grafo de proveniência do *workflow* científico modelado de forma conjunta

Mestrado em Modelagem Computacional





SciProv: A Scientific Workflow Provenance System

Semantic query on 'process' entities of the combined workflow provenance graph

fromProcess	fromValue	toProcess	toValue
id:p1	DDBJ Service	id:p5	Display Gene Sequence
id:p1	DDBJ Service	id:p4	Display Header
id:p1	DDBJ Service	id:p3	Extract Gene Sequence
id:p1	DDBJ Service	id:p2	Extract Reader
id:p2	Extract Reader	id:p4	Display Header
id:p3	Extract Gene Sequence	id:p5	Display Gene Sequence

[Home Page](#)

Figura 5.21: Resultado da busca semântica no *workflow* científico modelado de forma conjunta

lidade de definir-se especializações ao modelo OPM na forma de anotações, capazes de ampliar o processamento semântico em domínios particulares.

A arquitetura do *SciProv* provê o suporte a anotações segundo o OPM a partir do processo de instrumentalização de um *workflow* científico, em um procedimento análogo àquele apresentado na Seção 4.2.1 para as demais entidades e dependências causais do modelo. Também de forma coerente com a estratégia adotada no presente trabalho, as anotações capturadas durante a execução de um *workflow* instrumentalizado são persistentes na base de dados relacional do sistema.

Este estudo de caso tem por objetivo avaliar a aplicabilidade dos recursos fornecidos pelo arcabouço de anotações do OPM para permitir ao *SciProv* a composição de consultas semânticas com base nesse tipo de entidade do modelo.

A avaliação emprega o *workflow* científico modelado no SGWfC Taverna anteriormente utilizado na Seção 5.1.3. Especificamente, o objetivo consiste em acrescentar anotações às entidades do tipo Processo identificadas no experimento. Para efeito de estudo, é declarado um tipo de anotação no arcabouço denominado *hasPriority* cujo intento é acrescentar informação adicional sobre a importância ou prioridade de execução de cada entidade do tipo Processo para que o *workflow* possa ser considerado como corretamente ou completamente processado.

Esse tipo de informação é relevante em um cenário de experimentos científicos executados em uma grade computacional. Por exemplo, um serviço *web* utilizado no *workflow* pode torna-se indisponível durante a execução ou um processo pode retornar com erro. Assim, é interessante definir-se quais processos devem ser concluídos com sucesso para que o *workflow* modelado possa apresentar os resultados esperados.

A arquitetura do *SciProv* emprega o arcabouço de anotações do modelo OPM para configurar o tratamento de dois valores específicos para o tipo *hasPriority*. Na etapa de planejamento do estudo caso, os processos denominados *DDBJ Service* e *Extract Gene Sequence* receberam anotações do tipo *hasPriority* com o valor *High* e o processo *Extract Reader* recebeu anotação com o valor *Low*, conforme pode ser observado na Figura 5.22.

A partir do grafo de proveniência gerado pelo *SciProv* referente ao *workflow* científico modelado no SGWfC Taverna e instrumentalizado para a coleta dos metadados de proveniência — incluindo-se anotações —, torna-se possível elaborar consultas aos dados de linhagem com base na entidade de anotação OPM do tipo *hasPriority*. A Figura 5.23

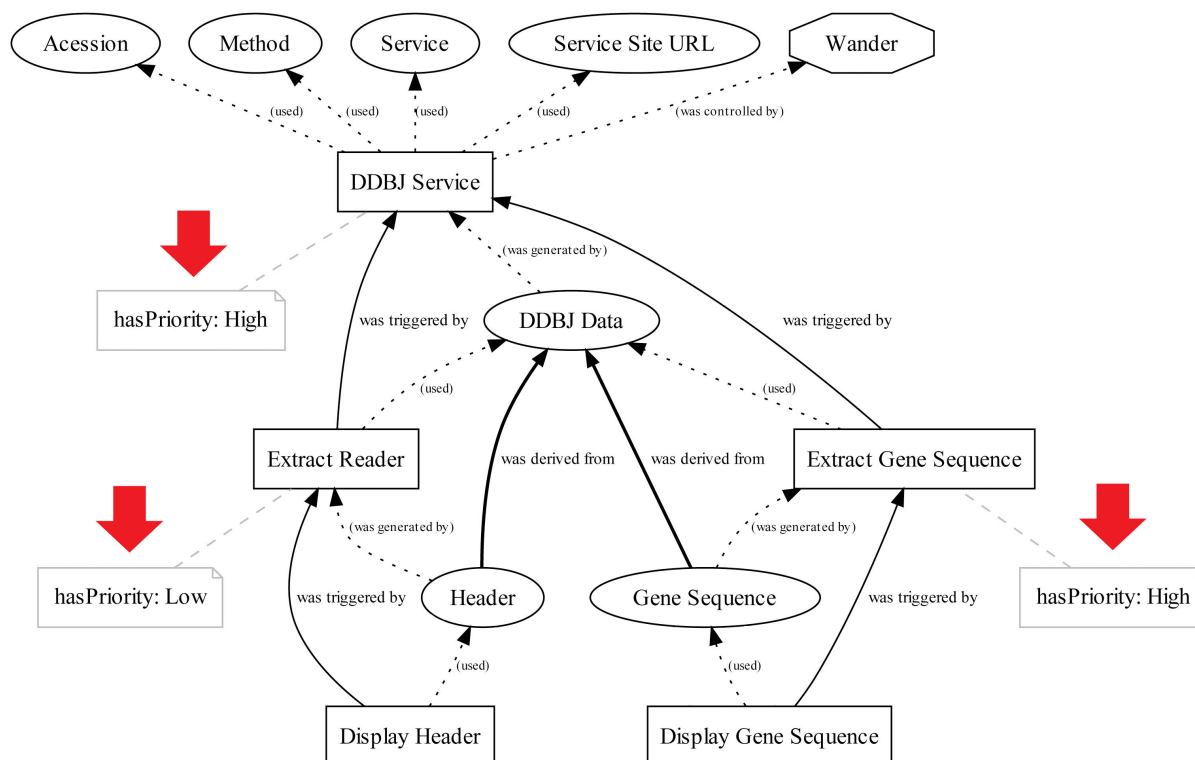


Figura 5.22: Grafo de proveniência com anotações OPM

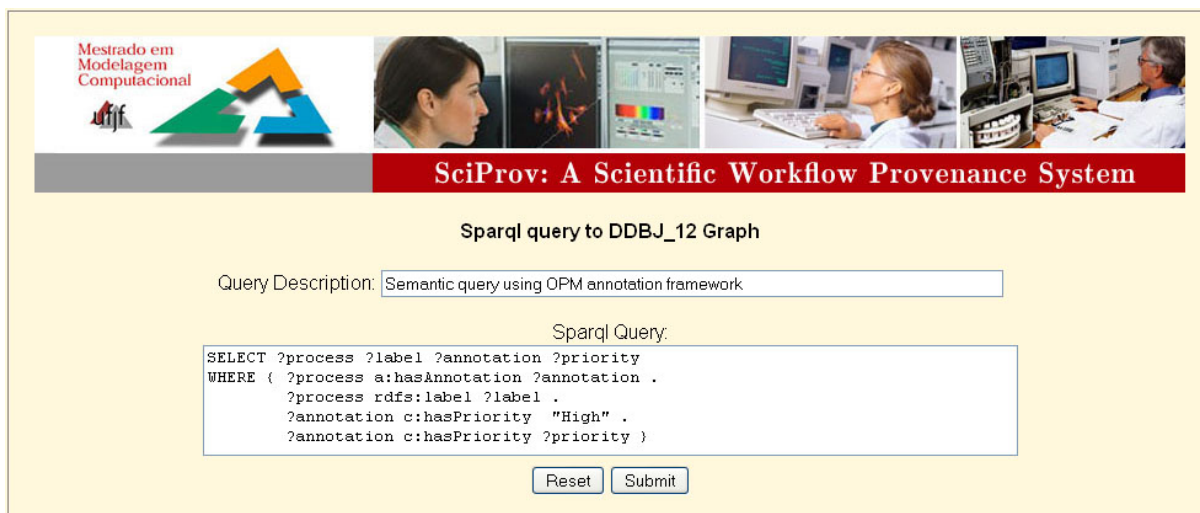
apresenta a sintaxe SPARQL para prover a pesquisa semântica cujo objetivo é retornar os processos cuja prioridade de execução esteja anotada como *High*.

Na etapa de avaliação, o resultado da consulta semântica, conforme apresentado na Figura 5.24, mostrou-se consistente com o grafo de proveniência exibido na Figura 5.22 ao retornar as entidades OPM do tipo Processo que foram anotadas com o valor *High* para a anotação *hasPriority*.

Ainda na etapa de avaliação de resultados, conforme apresentado neste estudo de caso, o acréscimo de anotações aos metadados de proveniência coletados segundo o modelo OPM pode enriquecer o conhecimento à cerca de experimentos científicos modelados computacionalmente. Nesse cenário, informações que não se encontram diretamente representadas em *workflows* científicos orquestrados em SGWfCs podem ser incorporadas ao grafo de proveniência e utilizadas para consultas semânticas capazes de extrair resultados além daqueles obtidos a partir das entidades e dependências causais padrão do modelo.

5.1.7 Estudo de caso com Agregação de Ontologia do Domínio

Além do emprego do arcabouço de anotações apresentado no estudo de caso anterior, é interessante avaliar ainda uma outra possibilidade de enriquecimento para o poder de



Mestrado em Modelagem Computacional

SciProv: A Scientific Workflow Provenance System

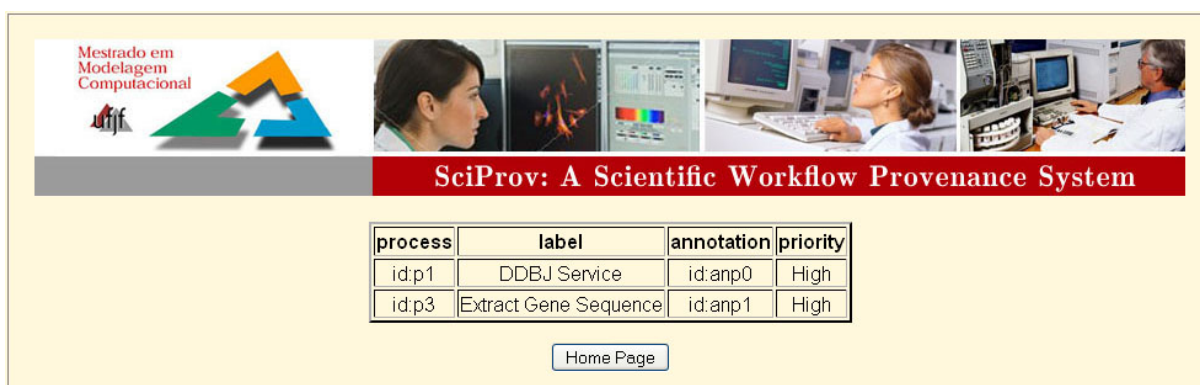
Sparql query to DDBJ_12 Graph

Query Description:

Sparql Query:

```
SELECT ?process ?label ?annotation ?priority
WHERE {
  ?process a:hasAnnotation ?annotation .
  ?process rdfs:label ?label .
  ?annotation c:hasPriority "High" .
  ?annotation c:hasPriority ?priority }
```

Figura 5.23: SPARQL para busca semântica com base em anotações OPM



Mestrado em Modelagem Computacional

SciProv: A Scientific Workflow Provenance System

process	label	annotation	priority
id.p1	DDBJ Service	id.anp0	High
id.p3	Extract Gene Sequence	id.anp1	High

Figura 5.24: Resultado da busca semântica com base em anotações OPM

processamento semântico do *SciProv*.

A arquitetura do sistema prevê o suporte à incorporação de informações de domínio do experimento científico modelado ao componente *TBox* do conhecimento representado no *SciProv*. Nesse contexto, conforme apresentado anteriormente na Seção 4.1.9 (vide Figura 4.12), é possível compor um *TBox* que reuna tanto a ontologia do modelo de proveniência OPM — acrescida da especialização para *workflows* científicos — quanto de uma segunda ontologia referente ao domínio da área científica em estudo.

Na etapa de planejamento, o presente estudo de caso agrega ao *TBox* do *SciProv* uma ontologia simples capaz de representar parcialmente a taxonomia do organismo *Synechococcus elongatus* PCC 7942 (Figura 5.25). Com base nessa ontologia incorporada ao *TBox* da arquitetura, buscou-se formular uma busca semântica ao grafo de proveniência para retornar a Família (*Synechococcaceae*) e o Gênero (*Synechococcus*) do organismo utilizado nos *workflows* científicos modelados no estudo de casos discutido nesse Capítulo.

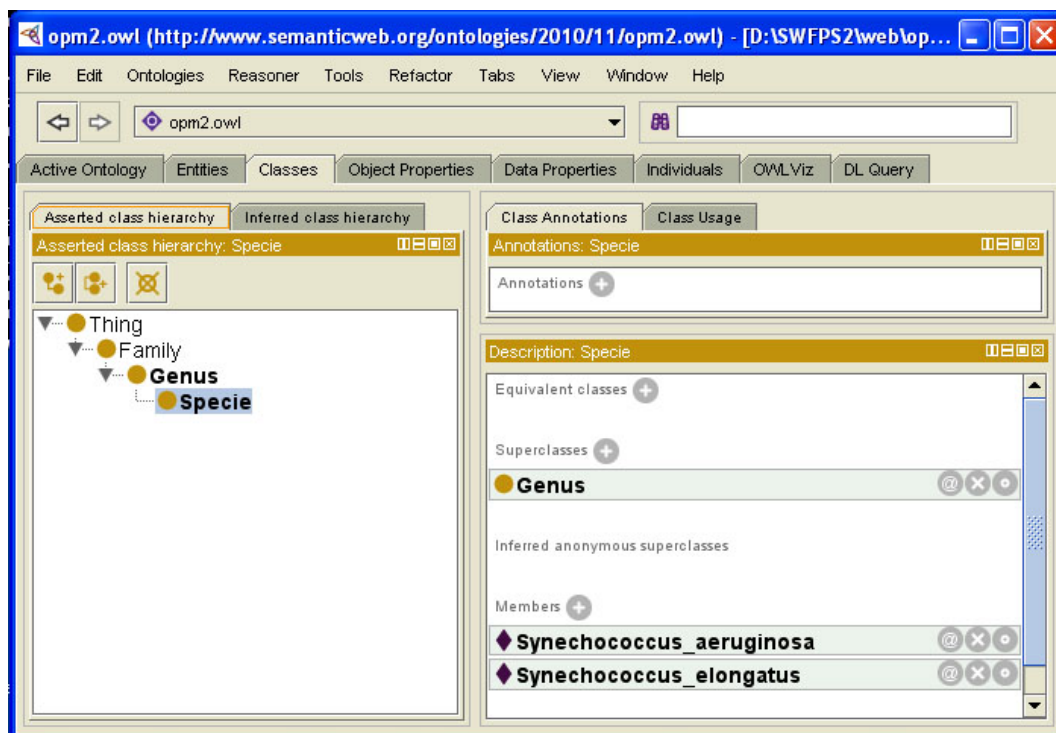


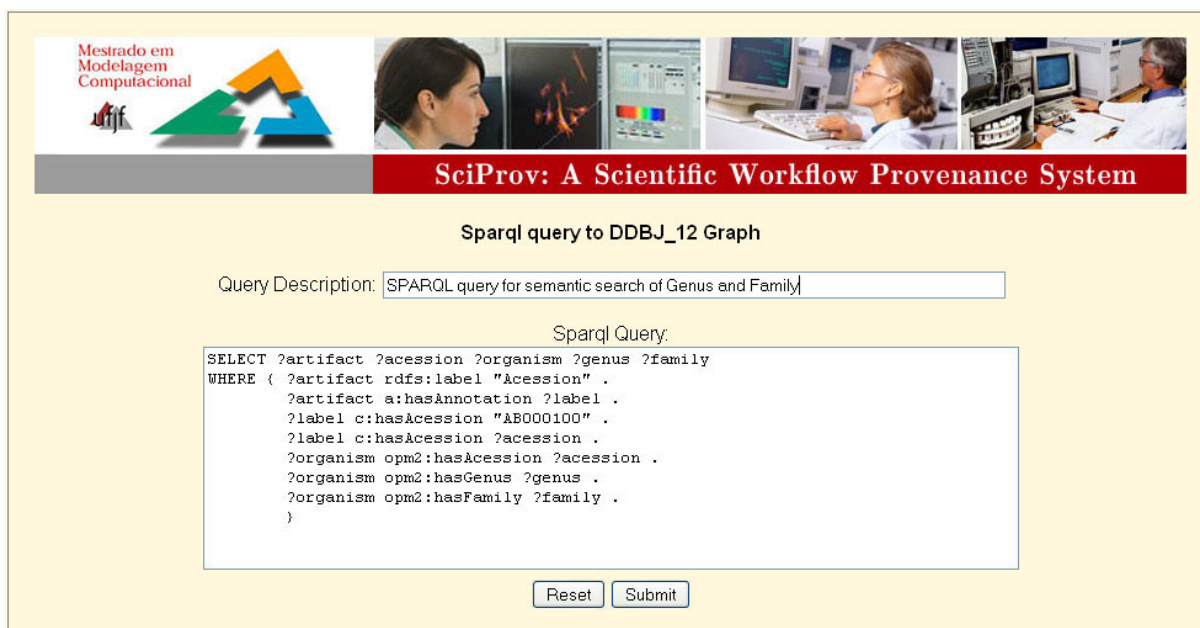
Figura 5.25: Ontologia simples referente à taxonomia parcial do organismo *Synechococcus elongatus*

A ontologia descreve unicamente a taxonomia parcial do Reino *Bacteria*, incluindo-se a Família *Synechococcaceare*, os Gêneros *Alternantia* e *Synechococcus*, e as Espécies *Synechococcus aeruginosa* e *Synechococcus elongatus* segundo a classificação de Cavalier-Smith [68].

Na etapa de monitoramento, a partir do *TBox* composto foi possível formular uma consulta semântica em SPARQL aos metadados de proveniência com o objetivo de avaliar a capacidade do *SciProv* de processar pesquisas com base em ontologias que incluam conhecimento sobre o domínio do experimento científico.

A Figura 5.26 apresenta uma consulta em SPARQL cujo propósito consistiu em fazer uma pesquisa semântica ao Gênero e à Família do organismo cujo código de acesso (*Accession*) foi fornecido como parâmetro de entrada para o serviço *web* do DDBJ utilizado no *workflow* científico modelado no SGWfC Taverna (ver Seção 5.1.3).

Como resultado da busca semântica ao grafo de proveniência, na etapa de avaliação dos resultados, a Figura 5.27 exhibe de forma consistente a Família e o Gênero do organismo *Synechococcus elongatus* PCC 7942 segundo a classificação de Cavalier-Smith. Esse estudo de caso mostra que a partir do *SciProv* é possível obter resultados adequados para pesquisas semânticas utilizando-se *TBox* compostos pela ontologia do modelo OPM em



Mestrado em Modelagem Computacional

SciProv: A Scientific Workflow Provenance System

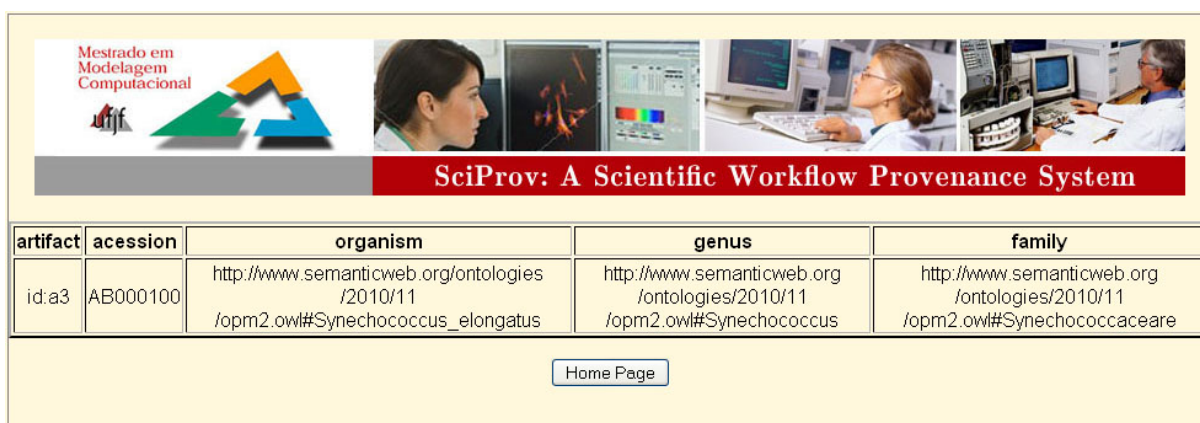
Sparql query to DDBJ_12 Graph

Query Description:

Sparql Query:

```
SELECT ?artifact ?acession ?organism ?genus ?family
WHERE {
?artifact rdfs:label "Acession" .
?artifact a:hasAnnotation ?label .
?label c:hasAcession "AB000100" .
?label c:hasAcession ?acession .
?organism opm2:hasAcession ?acession .
?organism opm2:hasGenus ?genus .
?organism opm2:hasFamily ?family .
}
```

Figura 5.26: Consulta SPARQL para busca semântica pelo Gênero e Família



Mestrado em Modelagem Computacional

SciProv: A Scientific Workflow Provenance System

artifact	acession	organism	genus	family
id:a3	AB000100	http://www.semanticweb.org/ontologies/2010/11/opm2.owl#Synechococcus_elongatus	http://www.semanticweb.org/ontologies/2010/11/opm2.owl#Synechococcus	http://www.semanticweb.org/ontologies/2010/11/opm2.owl#Synechococcaceare

Figura 5.27: Resultado da busca semântica por Gênero e Família

conjunto com ontologias que representam o domínio da área pesquisada.

Entretanto, cabe considerar que o processo de agregação de duas ou mais ontologias distintas ao *TBox* da arquitetura do *SciProv* pode requerer o processamento prévio de alinhamento de ontologias, cujo tratamento está além do escopo do presente trabalho.

6 CONSIDERAÇÕES FINAIS

O processo de concepção, execução e análise de experimentos científicos baseia-se cada vez mais na utilização de ferramentas computacionais. A informática representa um valioso recurso para o avanço da Ciência ao permitir a exploração de um grande volume de informação em muitos campos diferentes.

No entanto, os benefícios advindos implicam em novos e complexos desafios no cenário da pesquisa científica. Um aspecto importante refere-se à reutilização do conhecimento gerado. As experiências processadas em um ambiente de *e-Science* estão sujeitas a atualizações frequentes, seja por uma visão mais refinada dos pesquisadores ou em face de mudanças nos componentes e procedimentos. Em adição, bases de dados pode ser atualizadas periodicamente. Nesse contexto, a reutilização do conhecimento pode significar economia de tempo e recursos.

Outro aspecto relevante refere-se à análise de dados gerados, que pode ter a utilidade reduzida se os cientistas não conseguirem julgar a adequação dos resultados ao experimento em curso. Em um cenário de investigação científica, parte do significado dos dados deve-se ao entendimento do processo generativo.

Estas considerações conduzem à questão da proveniência de dados, que pode ser descrita como a história progressa de um item de dado, incluindo-se os processos pelo qual foi produzido. A proveniência de dados pode contribuir para a formação de uma visão de qualidade, validade e atualidade sobre um conjunto de informações produzido em um cenário de *e-Science*.

No entanto, para obter os benefícios oriundos do tratamento da proveniência de dados, é essencial que as informações relacionadas à simulação computacional que modela o experimento científico sejam capturadas e persistidas de alguma forma. Tal mecanismo deve ter acesso aos processos modelados em um *workflow* científico bem como aos dados gerados e às anotações.

Atualmente é possível tratar de forma isolada a linhagem dos dados gerados em *workflows* científicos modelados em SGWfCs como Vistrails, Kepler e Taverna. Mas a questão que se impõe refere-se a como gerir a proveniência de dados em um ambiente heterogêneo e distribuído. Nesse cenário, torna-se relevante propor-se um mecanismo capaz de

permitir a interoperabilidade entre os metadados de linhagem gerados.

O modelo de proveniência denominado *Open Proveniência Model* se apresenta como uma alternativa interessante nesse cenário por constituir-se em um conjunto de diretrizes para o apoio à linhagem em uma visão baseada em interoperabilidade, onde os metadados possam ser ligados de uma maneira coerente e consultados de forma integrada. Para tanto, OPM provê um vocabulário controlado, mecanismos de serialização em formatos da *web* semântica e APIs.

O emprego de tecnologia *web* semântica, um dos alicerces do modelo OPM, conduz a novas possibilidades para o tratamento dos metadados de proveniência. Em particular, oferece recursos para pesquisas semânticas utilizando-se de máquinas de inferência, que podem efetuar deduções sobre as bases de conhecimento representadas e obter resultados importantes ao extrair informações adicionais além daquelas que encontram-se registradas de forma explícita em grafos de proveniência.

O presente trabalho procura especificar uma arquitetura para a coleta e gerência da proveniência no contexto de experimentos científicos processados através de simulações computacionais em ambientes de pesquisa colaborativa dispersos geograficamente e interconectados através de uma grade computacional. A arquitetura proposta recebe o nome de *Scientific Workflow Provenance System*.

O *SciProv* define uma camada de interoperabilidade que interage com SGWfCs para capturar os metadados de proveniência gerados em *workflows* científicos processados computacionalmente. A arquitetura está baseada em requisitos considerados fundamentais para o desenvolvimento do trabalho, a saber:

- Independência em relação aos formatos de dados e processos empregados em SGWfCs.
- Aplicabilidade direcionada — mas não restrita — a experimentos científicos modelados computacionalmente em ambientes distribuídos e heterogêneos.
- Emprego de tecnologia *web* semântica — RDF, OWL, ontologias, máquinas de inferência — para representação e consulta aos metadados de proveniência.
- Possibilidade de adaptação do mecanismo de instrumentalização para permitir o controle sobre o impacto da coleta dos metadados no desempenho dos *workflows* científicos.

É possível observar na literatura científica propostas de arquiteturas para o tratamento da proveniência de dados no contexto de *e-Science*, porém o *SciProv* se diferencia por

apresentar um enfoque que se baseia no emprego abrangente de recursos da *web* semântica e na adoção do modelo abstrato OPM.

O *SciProv* adota uma solução baseada em serviços *web* para a coleta dos metadados de proveniência em razão de fatores como adequação à ambientes distribuídos e heterogêneos, uso de protocolos abertos e consensuais para a internet e interoperabilidade.

A arquitetura do *SciProv* permite a coleta e gerência da proveniência de dados em diversos níveis de abstração. Esse recurso pode ser considerado um importante diferencial em razão de diversos desdobramentos e potencialidades:

- Permite parametrizar o grafo de proveniência de acordo com o interesse da pesquisa em curso.
- Permite obter resultados com maior ou menor abrangência e detalhamento a partir de uma mesmo grafo de proveniência.
- Permite controlar o impacto do mecanismo de coleta sobre o desempenho de processamento ao habilitar apenas os níveis de abstração necessários em um determinado escopo de trabalho.
- Permite controlar o volume de metadados armazenados. Em um cenário onde *workflows* complexos têm potencial para processar grande quantidade de dados, é importante adequar a captura da proveniência aos recursos disponíveis.
- Facilita o processo de instrumentalização, ao permitir a modularização do trabalho.

Para a persistência dos metadados de proveniência coletados, o *SciProv* adota uma solução a partir de uma base de dados relacional em razão da confiabilidade e independência em relação a formatos de dados específicos utilizados por SGWfCs.

Em razão das implicações relacionadas a um modelo de armazenamento distribuído, o *SciProv* oferece uma solução centralizada para a persistência dos metadados coletados. Nessa abordagem simplificadora, a arquitetura emprega um único nó coordenador para o armazenamento dos metadados de proveniência.

Uma das características mais importantes do *SciProv* consiste em prover uma infraestrutura baseada na *web* semântica para gerenciar os metadados de proveniência. A arquitetura permite construir uma representação em memória aderente ao modelo OPM a partir dos metadados persistidos na base de dados relacional. Este grafo de proveniência modelado em memória permite a vinculação dos metadados em formatos compatíveis com a tecnologia *web* semântica e também a construção de uma representação gráfica

através de um diagrama que exhibe as entidades e dependências causais OPM. Embora tal representação gráfica não seja relevante no contexto da arquitetura, é interessante porque oferece ao cientista uma visão intuitiva da linhagem de um determinado dado produzido no contexto de um experimento científico.

O processo de vinculação do grafo de proveniência em formato RDF permite compor uma infraestrutura para a consulta aos metadados em SPARQL, a linguagem padrão recomendada pelo W3C no contexto da *web* semântica. Tanto a vinculação em RDF quanto a consulta em SPARQL estão associadas às ontologias do modelo OPM. Acrescenta-se ainda que o *SciProv* pode processar consultas semânticas a partir de um ou mais *TBoxes* incluídos na arquitetura. Nesse contexto, é possível anexar à ontologia do modelo OPM representações do conhecimento referentes ao domínio do *workflow* científico modelado.

A possibilidade de se construir novos perfis especializados ao modelo OPM é importante para que novos atributos semânticos possam ser incorporados ao grafo de proveniência. O *SciProv* promove uma especialização ao modelo OPM capaz de ampliar o poder de processamento semântico no domínio de *workflows* científicos. Este recurso constitui uma característica relevante ao potencializar a capacidade de inferência sobre os metadados de proveniência em um cenário de *e-Science*.

Outra possibilidade relevante de ampliação da capacidade de processamento semântico tratada pelo *SciProv* baseia-se no arcabouço de anotações do modelo OPM. Nesse cenário, o suporte a anotações provê um mecanismo para prover tanto uma maior interoperabilidade entre sistemas de proveniência quanto para a ampliação do conteúdo semântico.

O trabalho apresenta um conjunto de estudos de caso implementados com o objetivo de avaliar a adequabilidade do *SciProv* à coleta e gerência da proveniência no contexto de experimentos científicos processados em ambientes de pesquisa colaborativa dispersos geograficamente e interconectados através de uma grade computacional. Os estudos de caso empregam recursos computacionais na área de Bioinformática disponíveis no Banco de Dados de DNA do Japão e processados em um ambiente colaborativo do qual fazem parte instituições de pesquisa localizadas na Europa e nos Estados Unidos.

Além de demonstrar a aplicabilidade do *SciProv* em um cenário coerente com a arquitetura proposta, os estudos de caso modelados em dois SGWfCs distintos apresentam uma busca semântica ao grafo de proveniência. Este procedimento extrai informações que

não se encontram explicitamente representadas e que, portanto, são inferidas com base no conhecimento descrito na ontologia do modelo OPM especializada para o domínio de *workflows* científicos.

Um dos estudos de caso avalia o impacto da coleta e persistência dos metadados de proveniência sobre o desempenho dos experimentos científicos instrumentalizados pelo *SciProv*. São apresentadas análises comparativas simples referentes aos tempos de execução de *workflows* científicos adaptados para a captura e armazenamento dos metadados gerados.

Outro estudo de caso analisa uma situação relevante em um contexto de experimentação científica em ambientes colaborativos. Nesse contexto, é construído um grafo de proveniência a partir de um *workflow* modelado em conjunto utilizando-se dois SGWfCs distintos. Os resultados são consistentes com aqueles obtidos a partir dos grafos de proveniência gerados com base em *workflows* de igual funcionalidade construídos de forma independente.

São apresentados ainda estudos de caso para avaliação da busca semântica aos metadados de proveniência empregando-se o arcabouço de anotações do modelo OPM e com agregação de uma ontologia de domínio. Em ambas as avaliações, os resultados mostram-se interessantes e promissores. Mais importante, corroboram os objetivos propostos para a arquitetura do *SciProv*.

Com base nos estudos de casos apresentados, pode-se considerar promissores os resultados obtidos para a arquitetura do *SciProv*, avaliando-se os requisitos considerados fundamentais no desenvolvimento do trabalho. Merece destaque a possibilidade de extração de informações inferidas a partir de buscas semânticas e a capacidade de coletar e gerenciar os metadados de proveniência de forma independente de formatos específicos de SGWfC.

O tratamento da proveniência de dados constitui um tema de pesquisa em aberto e, no contexto de *e-Science*, algumas questões podem ser colocadas como sugestões para trabalhos futuros relacionados à arquitetura do *SciProv*.

- Refinamento no mecanismo de especialização da ontologia OPM para o domínio de *workflows* científicos, de forma a representar de forma mais consistente e abrangente as particularidades de um grafo de proveniência. Como consequência, as buscas semânticas poderão retornar resultados mais significativos, ao processar inferências

sobre uma representação do conhecimento mais sofisticada.

- Refinamento no mecanismo de instrumentalização de *workflows* científicos, com o objetivo de otimizar o desempenho do processo de coleta e persistência dos metadados de proveniência. Nesse aspecto, pode-se cogitar desde uma alternativa mais simples para estudo posterior, como o envio em lotes dos metadados coletados para o repositório centralizado, até uma alternativa mais complexa, referente ao armazenamento distribuído da linhagem em nós distintos da grade computacional do ambiente colaborativo.
- Avaliação de tecnologias alternativas para a consulta semântica à proveniência. Pode-se argumentar que a linguagem SPARQL, embora considerada padrão pelo W3C para pesquisa nesse contexto, impõe limitações ao uso do *SciProv* por parte da comunidade científica proposta em razão da complexidade. Uma hipótese interessante porém complexa a ser considerada consiste em utilizar a representação gráfica do grafo de proveniência para formular consultas a partir de uma interface visual e intuitiva.
- Tratamento de uma questão não contemplada na arquitetura do *SciProv* referente à busca semântica a partir de alinhamento de ontologias. Esse recurso pode ser promissor ao permitir o processamento de inferências em representações do conhecimento que incluam não só o modelo de proveniência OPM como também ontologias específicas sobre o tema científico em estudo. O presente trabalho apresenta um estudo de caso bastante simples nesse contexto, mas sugere que o tema seja tratado de forma específica.
- Manter uma aderência continuada da arquitetura proposta ao modelo abstrato OPM, que encontra-se em rápido desenvolvimento e atualização. O modelo OPM tem procurado responder aos diversos desafios impostos ao tratamento da proveniência de dados. Nesse contexto, é importante que o *SciProv* seja capaz de incorporar as funcionalidades do modelo que vierem a ser incluídas ou aprimoradas.

REFERÊNCIAS

- [1] MATTOSO, M., WERNER, C., TRAVASSOS, G., BRAGANHOLO, V., MURTA, L., “Gerenciando experimentos científicos em larga escala”. In: *XXVIII Seminário Integrado de Software e Hardware*, pp. 121–135, Sociedade Brasileira de Computação: Porto Alegre, RS, Brazil, 2008.
- [2] BUNEMAN, P., KHANNA, S., TAN, W. C., “Why and Where: A Characterization of Data Provenance”. In: *ICDT '01: Proceedings of the 8th International Conference on Database Theory*, pp. 316–330, Springer-Verlag: London, UK, 2001.
- [3] SIMMHAN, Y. L., PLALE, B., GANNON, D., “A survey of data provenance in e-science”, *SIGMOD Rec.*, v. 34, n. 3, pp. 31–36, 2005.
- [4] GROTH, P., DEELMAN, E., JUVE, G., MEHTA, G., BERRIMAN, B., “Pipeline-centric provenance model”. In: *WORKS '09: Proceedings of the 4th Workshop on Workflows in Support of Large-Scale Science*, pp. 1–8, ACM: New York, NY, USA, 2009.
- [5] GODERIS, A., SATTLER, U., LORD, P. W., GOBLE, C. A., “Seven Bottlenecks to Workflow Reuse and Repurposing”. In: *International Semantic Web Conference*, pp. 323–337, 2005.
- [6] MUNROE, S., MILES, S., MOREAU, L., VÁZQUEZ-SALCEDA, J., “PrIME: a software engineering methodology for developing provenance-aware applications”. In: *SEM '06: Proceedings of the 6th international workshop on Software engineering and middleware*, pp. 39–46, ACM: New York, NY, USA, 2006.
- [7] SIMMHAN, Y. L., PLALE, B., GANNON, D., “A Framework for Collecting Provenance in Data-Centric Scientific Workflows”. In: *ICWS '06: Proceedings of the IEEE International Conference on Web Services*, pp. 427–436, IEEE Computer Society: Washington, DC, USA, 2006.
- [8] BITON, O., COHEN-BOULAKIA, S., DAVIDSON, S. B., HARA, C. S., “Querying and Managing Provenance through User Views in Scientific Workflows”. In:

ICDE '08: Proceedings of the 2008 IEEE 24th International Conference on Data Engineering, pp. 1072–1081, IEEE Computer Society: Washington, DC, USA, 2008.

- [9] FREIRE, J., KOOP, D., SANTOS, E., SILVA, C. T., “Provenance for Computational Tasks: A Survey”, *Computing in Science and Engineering*, v. 10, n. 3, pp. 11–21, 2008.
- [10] MARINHO, A., MURTA, L., WERNER, C., BRAGANHOLO, V., DA CRUZ, S. M. S., OGASAWARA, E., MATTOSO, M., “Managing Provenance in Scientific Workflows with ProvManager”. In: *22nd International Symposium on Computer Architecture and High Performance Computing*, 2010.
- [11] MOREAU, L., FREIRE, J., FUTRELLE, J., MCGRATH, R. E., MYERS, J., PAULSON, P., “The Open Provenance Model: An Overview”, , pp. 323–326 2008.
- [12] FREIRE, J., SILVA, C. T., CALLAHAN, S. P., SANTOS, E., SCHEIDEGGER, C. E., VO, H. T., “Managing rapidly-evolving scientific workflows”. In: *International Provenance and Annotation Workshop (IPAW), LNCS 4145*, pp. 10–18, Springer Verlag, 2006.
- [13] LUDÄSCHER, B., ALTINTAS, I., BERKLEY, C., HIGGINS, D., JAEGER, E., JONES, M., LEE, E. A., TAO, J., ZHAO, Y., “Scientific workflow management and the Kepler system: Research Articles”, *Concurr. Comput. : Pract. Exper.*, v. 18, n. 10, pp. 1039–1065, 2006.
- [14] ZHAO, J., GOBLE, C., STEVENS, R., TURI, D., “Mining Taverna’s semantic web of provenance”, *Concurr. Comput. : Pract. Exper.*, v. 20, n. 5, pp. 463–472, 2008.
- [15] MATTOSO, M., WERNER, C., TRAVASSOS, G., BRAGANHOLO, V., MURTA, L., OGASAWARA, E., MARTINHO, W., “Desafios no apoio a composição de experimentos científicos em larga escala”. In: *XXXVI Seminário Integrado de Software e Hardware*, pp. 307–321, Sociedade Brasileira de Computação: Porto Alegre, RS, Brazil, 2009.

- [16] CRUZ, S. M. S. D., CAMPOS, M. L. M., MATTOSO, M., “Towards a Taxonomy of Provenance in Scientific Workflow Management Systems”. In: *SERVICES '09: Proceedings of the 2009 Congress on Services - I*, pp. 259–266, IEEE Computer Society: Washington, DC, USA, 2009.
- [17] MATOS, E. E., CAMPOS, F., BRAGA, R., PALAZZI, D., “CelOWS: An ontology based framework for the provision of semantic web services related to biological models”, *J. of Biomedical Informatics*, v. 43, pp. 125–136, February 2010.
- [18] MENDES, L. F., BRAGA, R., CAMPOS, F., “SASAgent: an agent based architecture for search, retrieval and composition of e-science models and tools”. In: *Proceedings of the First international conference on Information technology in bio- and medical informatics, ITBAM'10*, pp. 86–93, Springer-Verlag: Berlin, Heidelberg, 2010.
- [19] PALAZZI, D., MATOS, E. E., CAMPOS, F., BRAGA, R., “Development Approach for e-Science Ontology: A Case Study in Biological Domain”. In: *Proceedings of the 2010 14th IEEE International Enterprise Distributed Object Computing Conference Workshops, EDOCW '10*, pp. 293–300, IEEE Computer Society: Washington, DC, USA, 2010.
- [20] LUCENA, C., MEDEIROS, C., LUCCHESI, C., MALDONADO, J., ALMEIDA, V., “Grandes Desafios da Pesquisa em Computação no Brasil: 2006-2016”, 2006.
- [21] MENAGER, H., LACROIX, Z., “A Workflow Engine for the Execution of Scientific Protocols”. In: *ICDEW '06: Proceedings of the 22nd International Conference on Data Engineering Workshops*, p. 68, IEEE Computer Society: Washington, DC, USA, 2006.
- [22] OGASAWARA, E., DE OLIVEIRA, D., CHIRIGATI, F., BARBOSA, C. E., ELIAS, R., BRAGANHOLO, V., COUTINHO, A., MATTOSO, M., “Exploring many task computing in scientific workflows”. In: *MTAGS '09: Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers*, pp. 1–10, ACM: New York, NY, USA, 2009.
- [23] ALTINTAS, I., “Lifecycle of Scientific Workflows and their Provenance: A Usage Perspective”. In: *SERVICES '08: Proceedings of the 2008 IEEE Congress on*

Services - Part I, pp. 474–475, IEEE Computer Society: Washington, DC, USA, 2008.

- [24] COGBURN, D. L., “HCI in the so-called developing world: what’s in it for everyone”, *interactions*, v. 10, n. 2, pp. 80–87, 2003.
- [25] CHIN, JR., G., LANSING, C. S., “Capturing and supporting contexts for scientific data sharing via the biological sciences collaboratory”. In: *CSCW ’04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pp. 409–418, ACM: New York, NY, USA, 2004.
- [26] FOSTER, I., KESSELMAN, C., (eds), “The grid”, chap. The Globus toolkit, pp. 259–278, Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 1999.
- [27] MOREAU, L., GROTH, P., ZHAO, J., “The Open Provenance Model”. In: *FIS’ 10: Future Internet Symposium*, STI International: Berlin, Germany, 2010.
- [28] HORTA, G., BARROS, M., “Contributions of In Virtuo and In Silico Experiments for the Future of Empirical Studies in Software Engineering”. In: *WSESE’ 03: 2nd Workshop on Empirical Software Engineering: The Future of Empirical Studies in Software Engineering*, pp. 427–436, Fraunhofer IRB Verlag: Roma, Italy, 2003.
- [29] GIL, Y., DEELMAN, E., ELLISMAN, M., FAHRINGER, T., FOX, G., GANNON, D., GOBLE, C., LIVNY, M., MOREAU, L., MYERS, J., “Examining the Challenges of Scientific Workflows”, *Computer*, v. 40, n. 12, pp. 24–32, 2007.
- [30] OINN, T., LI, P., KELL, D. B., GOBLE, C., GODERIS, A., GREENWOOD, M., HULL, D., STEVENS, R., TURI, D., ZHAO, J., “Taverna/myGrid: Aligning a Workflow System with the Life Sciences Community”, In: *Workflows for e-Science: Scientific workflows for Grids*, chap. part III, pp. 300–319, Springer London, 2007.
- [31] BOSE, R., FREW, J., “Lineage retrieval for scientific data processing: a survey”, *ACM Comput. Surv.*, v. 37, n. 1, pp. 1–28, 2005.
- [32] DAVIDSON, S. B., FREIRE, J., “Provenance and scientific workflows: challenges and opportunities”. In: *SIGMOD ’08: Proceedings of the 2008 ACM SIGMOD*

international conference on Management of data, pp. 1345–1350, ACM: New York, NY, USA, 2008.

- [33] GROTH, P., JIANG, S., MILES, S., MUNROE, S., TAN, V., TSASAKOU, S., MOREAU, L., *An architecture for provenance systems*, Tech. Rep. D3.1.1 Final Architecture v.0.6, EU Provenance Project, 2006.
- [34] DIGIAMPIETRI, L., MEDEIROS, C., SETÚBAL, J., “A framework based in Web services orchestration bioinformatics workflow management”, *Genetics and Molecular Research*, v. 4, n. 3, pp. 535–542, 2005.
- [35] GROTH, P., MILES, S., MOREAU, L., “PReServ: Provenance Recording for Services”. In: *In Proceedings of the UK OST e-Science second All Hands Meeting 2005 (AHM’05), Nottingham, UK*, 2005.
- [36] FREW, J., METZGER, D., SLAUGHTER, P., “Automatic capture and reconstruction of computational provenance”, *Concurr. Comput. : Pract. Exper.*, v. 20, n. 5, pp. 485–496, 2008.
- [37] MUNISWAMY-REDDY, K.-K., HOLLAND, D. A., BRAUN, U., SELTZER, M., “Provenance-aware storage systems”. In: *ATEC ’06: Proceedings of the annual conference on USENIX ’06 Annual Technical Conference*, pp. 4–4, USENIX Association: Berkeley, CA, USA, 2006.
- [38] BARGA, R. S., DIGIAMPIETRI, L. A., “Automatic capture and efficient storage of e-Science experiment provenance”, *Concurr. Comput. : Pract. Exper.*, v. 20, n. 5, pp. 419–429, 2008.
- [39] KLOSS, G. K., SCHREIBER, A., “Provenance Implementation in a Scientific Simulation Environment”. In: *International Provenance and Annotation Workshop (I-PAW), 2006, SpringerVerlag LNCS*, pp. 28–37, Springer-Verlag, 2006.
- [40] KIFOR, T., VARGA, L., VAZQUEZ-SALCEDA, J., ALVAREZ, S., WILLMOTT, S., *EHCR: An EU Provenance Case Study*, Tech. rep., EU Provenance Project, 2006.
- [41] MOREAU, L., IBBOTSON, J., *Standardisation of Provenance Systems in Service Oriented Architectures*, Tech. Rep. 12198, University of Southampton, 2006.

- [42] COHEN, S., COHEN-BOULAKIA, S., DAVIDSON, S., “Towards a model of provenance and user views in scientific workflows”. In: *In Data Integration in the Life Sciences*, pp. 264–279, 2006.
- [43] SCHEIDEGGER, C., KOOP, D., SANTOS, E., VO, H., CALLAHAN, S., FREIRE, J., SILVA, C., “Tackling the Provenance Challenge one layer at a time”, *Concurr. Comput. : Pract. Exper.*, v. 20, n. 5, pp. 473–483, 2008.
- [44] KIM, J., DEELMAN, E., GIL, Y., MEHTA, G., RATNAKAR, V., “Provenance trails in the Wings-Pegasus system”, *Concurr. Comput. : Pract. Exper.*, v. 20, pp. 587–597, April 2008.
- [45] MOREAU, L., GROTH, P., MILES, S., VAZQUEZ, J., IBBOTSON, J., JIANG, S., MUNROE, S., RANA, O., SCHREIBER, A., TAN, V., VARGA, L., “The Provenance of Electronic Data”, *Communications of the ACM*, v. 51, n. 4, pp. 52–58, 2008.
- [46] ALMEIDA, M. B., BAX, M. P., “Uma visão geral sobre ontologias: pesquisa sobre definições, tipos, aplicações, métodos de avaliação e de construção”, *Ciência da Informação*, v. 32, n. 3, pp. 7–20, 2003.
- [47] BREITAM, K. K., *Web Semântica: a Internet do Futuro*. LTC: Rio de Janeiro, RJ, Brasil, 2005.
- [48] ROURE, D., GOBLE, C., “Supporting e-Science Using Semantic Web Technologies – The Semantic Grid”, In: *Semantic e-Science*, v. 11, pp. 1–28, *Annals of Information Systems*, Springer US, 2010.
- [49] GOLBECK, J., HENDLER, J., “A Semantic Web approach to the provenance challenge”, *Concurr. Comput. : Pract. Exper.*, v. 20, n. 5, pp. 431–439, 2008.
- [50] ZEDNIK, S., FOX, P., MCGUINNESS, D., SILVA, P., CHANG, C., “Semantic Provenance for Science Data Products: Application to Image Data Processing”. In: *Workshop on Semantic Web and Provenance Management*, v. 526, *CEUR Workshop Proceedings*, CEUR-WS.org, 2008.

- [51] FUTRELLE, J., MYERS, J., “Tracking provenance semantics in heterogeneous execution systems”, *Concurr. Comput. : Pract. Exper.*, v. 20, n. 5, pp. 555–564, 2008.
- [52] OLSON, G. M., “The next generation of science collaboratories”. In: *CTS '09: Proceedings of the 2009 International Symposium on Collaborative Technologies and Systems*, pp. xv–xvi, IEEE Computer Society: Washington, DC, USA, 2009.
- [53] FREIRE, J., SILVA, C. T., “Towards enabling social analysis of scientific data”. In: *In CHI Social Data Analysis Workshop*, Florence, Italy, 2008.
- [54] MOREAU, L., CLIFFORD, B., FREIRE, J., FUTRELLE, J., GIL, Y., GROTH, P., KWASNIKOWSKA, N., MILES, S., MISSIER, P., MYERS, J., PLALE, B., SIMMHAN, Y., STEPHAN, E., DEN BUSSCHE, J. V., “The Open Provenance Model core specification (v1.1)”, *Future Generation Computer Systems*, v. In Press, Corrected Proof, pp. –, 2010.
- [55] OZSU, M. T., *Principles of Distributed Database Systems*. Prentice Hall Press: Upper Saddle River, NJ, USA, 2007.
- [56] MOREAU, L., FREIRE, J., FUTRELLE, J., MYERS, J., PAULSON, P., “Governance of the Open Provenance Model”, 2009.
- [57] GANSNER, E. R., KOUTSOFIOS, E., NORTH, S. C., VO, K.-P., “A Technique for Drawing Directed Graphs”, *IEEE Trans. Softw. Eng.*, v. 19, n. 3, pp. 214–230, 1993.
- [58] BAADER, F., CALVANESE, D., MCGUINNESS, D. L., NARDI, D., PATEL-SCHNEIDER, P. F., *The Description Logic Handbook: Theory, Implementation and Applications*. 2nd ed. Cambridge University Press: New York, NY, USA, 2010.
- [59] PAN, J. Z., “OWL for the Novice - A Logical Perspective”, In: *Semantic Web: Revolutionizing Knowledge Discovery in the Life Sciences*, chap. Chapter 8, pp. 159–182, Springer, 2007.
- [60] ZHAO, J., “Open Provenance Model Vocabulary Specification”, Revision: 0.4.

- [61] FUTRELLE, J., GAYNOR, J., PLUTCHAK, J., MYERS, J. D., MCGRATH, R. E., BAJCSY, P., KASTNER, J., KOTWANI, K., LEE, J. S., MARINI, L., KOOPER, R., MCLAREN, T., LIU, Y., “Semantic middleware for e-science knowledge spaces”. In: *MGC '09: Proceedings of the 7th International Workshop on Middleware for Grids, Clouds and e-Science*, pp. 1–6, ACM: New York, NY, USA, 2009.
- [62] BROEKSTRA, J., KAMPMAN, A., HARMELEN, F. V., “Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema”. In: *ISWC '02: Proceedings of the First International Semantic Web Conference on The Semantic Web*, pp. 54–68, Springer-Verlag: London, UK, 2002.
- [63] CARROLL, J. J., DICKINSON, I., DOLLIN, C., REYNOLDS, D., SEABORNE, A., WILKINSON, K., “Jena: implementing the semantic web recommendations”. In: *WWW Alt. '04: Proceedings of the 13th International World Wide Web Conference*, pp. 74–83, ACM: New York, NY, USA, 2004.
- [64] SIRIN, E., PARSIA, B., GRAU, B. C., KALYANPUR, A., KATZ, Y., “Pellet: A practical OWL-DL reasoner”, *Web Semant.*, v. 5, n. 2, pp. 51–53, 2007.
- [65] KITCHENHAM, B., PICKARD, L., PFLEEGER, S. L., “Case Studies for Method and Tool Evaluation”, *IEEE Softw.*, v. 12, pp. 52–62, July 1995.
- [66] TATENO, Y., IMANISHI, T., MIYAZAKI, S., FUKAMI-KOBAYASHI, K., SAITOU, N., SUGAWARA, H., GOJOBORI, T., “DNA Data Bank of Japan (DDBJ) for genome scale research in life science”, *Nucleic Acids Research*, v. 30, n. 1, pp. 27–30, 2002.
- [67] SCHIKUTA, E., WANEEK, H., UL HAQ, I., “Grid workflow optimization regarding dynamically changing resources and conditions”, *Concurr. Comput. : Pract. Exper.*, v. 20, pp. 1837–1849, October 2008.
- [68] CAVALIER-SMITH, T., “Only six kingdoms of life.” *Proceedings of the Royal Society B Biological Sciences*, v. 271, n. 1545, pp. 1251–1262, 2004.

APÊNDICE A - Ontologia do Modelo OPM Especializada no Domínio de Workflows Científicos

```

<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
  <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
  <!ENTITY opm "http://www.ipaw.info/2007/opm#" >
  <!ENTITY swrl "http://www.w3.org/2003/11/swrl#" >
  <!ENTITY swrlb "http://www.w3.org/2003/11/swrlb#" >
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
  <!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
  <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
  <!ENTITY protege "http://protege.stanford.edu/plugins/owl/protege#" >
  <!ENTITY xsp "http://www.owl-ontologies.com/2005/08/07/xsp.owl#" >
]>

<rdf:RDF xmlns="http://www.ipaw.info/2007/opm#"
  xml:base="http://www.ipaw.info/2007/opm"
  xmlns:opm="http://www.ipaw.info/2007/opm#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:swrl="http://www.w3.org/2003/11/swrl#"
  xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
  xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
  xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"

```

```

xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
<owl:Ontology rdf:about=""/>

<!--
////////////////////////////////////
//
// Annotation properties
//
////////////////////////////////////
-->

<owl:AnnotationProperty rdf:about="&rdfs;comment"/>

<!--
////////////////////////////////////
//
// Object Properties
//
////////////////////////////////////
-->

<!-- http://www.ipaw.info/2007/opm#controlledAccount -->

<owl:ObjectProperty rdf:about="#controlledAccount">
  <rdfs:range rdf:resource="#Account"/>
  <rdfs:domain rdf:resource="#Controlled"/>
  <rdfs:subPropertyOf rdf:resource="#eventAccount"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#controlledByAgent -->

<owl:ObjectProperty rdf:about="#controlledByAgent">

```

```

    <rdfs:comment rdf:datatype="&xsd:string"
      >the agent that controlled the process</rdfs:comment>
    <rdfs:range rdf:resource="#Agent"/>
    <rdfs:domain rdf:resource="#Controlled"/>
  </owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#controlledProcess -->

<owl:ObjectProperty rdf:about="#controlledProcess">
  <rdfs:comment rdf:datatype="&xsd:string"
    >the process that the agent controlled</rdfs:comment>
  <rdfs:domain rdf:resource="#Controlled"/>
  <rdfs:range rdf:resource="#Process"/>
  <rdfs:subPropertyOf rdf:resource="#eventProcess"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#controlledRole -->

<owl:ObjectProperty rdf:about="#controlledRole">
  <rdfs:comment rdf:datatype="&xsd:string"
    >the role the agent played in controlling the process
    </rdfs:comment>
  <rdfs:domain rdf:resource="#Controlled"/>
  <rdfs:range rdf:resource="#Role"/>
  <rdfs:subPropertyOf rdf:resource="#eventRole"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#controlledStartTime -->

<owl:ObjectProperty rdf:about="#controlledStartTime">
  <rdfs:comment rdf:datatype="&xsd:string"
    >the time at which the agent began controlling the process

```



```

        </rdfs:comment>
        <rdfs:domain rdf:resource="#Controlled"/>
        <rdfs:subPropertyOf rdf:resource="#eventTime"/>
    </owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#controlledTerminateTime -->

<owl:ObjectProperty rdf:about="#controlledTerminateTime">
    <rdfs:comment rdf:datatype="&xsd:string"
        >the time at which the agent ceased controlling the process
    </rdfs:comment>
    <rdfs:domain rdf:resource="#Controlled"/>
    <rdfs:subPropertyOf rdf:resource="#eventTime"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#derivedAccount -->

<owl:ObjectProperty rdf:about="#derivedAccount">
    <rdfs:range rdf:resource="#Account"/>
    <rdfs:domain rdf:resource="#Derived"/>
    <rdfs:subPropertyOf rdf:resource="#eventAccount"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#derivedArtifact -->

<owl:ObjectProperty rdf:about="#derivedArtifact">
    <rdf:type rdf:resource="&owl;TransitiveProperty"/>
    <rdfs:comment rdf:datatype="&xsd:string"
        >the artifact that was derived</rdfs:comment>
    <rdfs:range rdf:resource="#Artifact"/>
    <rdfs:domain rdf:resource="#Derived"/>
    <owl:inverseOf rdf:resource="#derivedFromArtifact"/>

```

```

    <rdfs:subPropertyOf rdf:resource="#eventArtifact"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#derivedFromArtifact -->

<owl:ObjectProperty rdf:about="#derivedFromArtifact">
  <rdf:type rdf:resource="#owl:TransitiveProperty"/>
  <rdfs:comment rdf:datatype="xsd:string"
    >the artifact from which the derived artifact was derived
  </rdfs:comment>
  <rdfs:range rdf:resource="#Artifact"/>
  <rdfs:domain rdf:resource="#Derived"/>
  <rdfs:subPropertyOf rdf:resource="#eventArtifact"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#derivedTime -->

<owl:ObjectProperty rdf:about="#derivedTime">
  <rdfs:comment rdf:datatype="xsd:string"
    >the time at which the artifact was derived</rdfs:comment>
  <rdfs:domain rdf:resource="#Derived"/>
  <rdfs:subPropertyOf rdf:resource="#eventTime"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#eventAccount -->

<owl:ObjectProperty rdf:about="#eventAccount">
  <rdfs:comment rdf:datatype="xsd:string"
    >an association between an event and an account</rdfs:comment>
  <rdfs:range rdf:resource="#Account"/>
  <rdfs:domain rdf:resource="#Event"/>
</owl:ObjectProperty>

```

```
<!-- http://www.ipaw.info/2007/opm#eventArtifact -->

<owl:ObjectProperty rdf:about="#eventArtifact">
  <rdfs:comment rdf:datatype="&xsd:string"
    >an association between an event and an artifact</rdfs:comment>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#eventProcess -->

<owl:ObjectProperty rdf:about="#eventProcess">
  <rdfs:comment rdf:datatype="&xsd:string"
    >an association between an event and a process</rdfs:comment>
  <rdfs:domain rdf:resource="#Event"/>
  <rdfs:range rdf:resource="#Process"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#eventRole -->

<owl:ObjectProperty rdf:about="#eventRole">
  <rdfs:comment rdf:datatype="&xsd:string"
    >the role associated with an event, e.g., for what was an
    artifact used?</rdfs:comment>
  <rdfs:domain rdf:resource="#Event"/>
  <rdfs:range rdf:resource="#Role"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#eventTime -->

<owl:ObjectProperty rdf:about="#eventTime">
  <rdfs:comment rdf:datatype="&xsd:string"
    >a piece of timing information associated with an event
```

```

        </rdfs:comment>
        <rdfs:domain rdf:resource="#Event"/>
        <rdfs:range rdf:resource="#TimeInterval"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#generatedAccount -->

<owl:ObjectProperty rdf:about="#generatedAccount">
    <rdfs:domain rdf:resource="#Generated"/>
    <rdfs:subPropertyOf rdf:resource="#eventAccount"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#generatedArtifact -->

<owl:ObjectProperty rdf:about="#generatedArtifact">
    <rdfs:comment rdf:datatype="&xsd:string"
        >the artifact that was generated</rdfs:comment>
    <rdfs:range rdf:resource="#Artifact"/>
    <rdfs:domain rdf:resource="#Generated"/>
    <rdfs:subPropertyOf rdf:resource="#eventArtifact"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#generatedByProcess -->

<owl:ObjectProperty rdf:about="#generatedByProcess">
    <rdfs:comment rdf:datatype="&xsd:string"
        >the process that generated the artifact</rdfs:comment>
    <rdfs:domain rdf:resource="#Generated"/>
    <rdfs:range rdf:resource="#Process"/>
    <rdfs:subPropertyOf rdf:resource="#eventProcess"/>
</owl:ObjectProperty>

```

```
<!-- http://www.ipaw.info/2007/opm#generatedRole -->
```

```
<owl:ObjectProperty rdf:about="#generatedRole">
```

```
  <rdfs:comment rdf:datatype="&xsd:string"
```

```
    >the role the generated artifact serves with respect to  
    the process</rdfs:comment>
```

```
  <rdfs:domain rdf:resource="#Generated"/>
```

```
  <rdfs:range rdf:resource="#Role"/>
```

```
  <rdfs:subPropertyOf rdf:resource="#eventRole"/>
```

```
</owl:ObjectProperty>
```

```
<!-- http://www.ipaw.info/2007/opm#generatedTime -->
```

```
<owl:ObjectProperty rdf:about="#generatedTime">
```

```
  <rdfs:comment rdf:datatype="&xsd:string"
```

```
    >the time at which the artifact was generated</rdfs:comment>
```

```
  <rdfs:domain rdf:resource="#Generated"/>
```

```
  <rdfs:subPropertyOf rdf:resource="#eventTime"/>
```

```
</owl:ObjectProperty>
```

```
<!-- http://www.ipaw.info/2007/opm#triggeredAccount -->
```

```
<owl:ObjectProperty rdf:about="#triggeredAccount">
```

```
  <rdfs:domain rdf:resource="#Triggered"/>
```

```
  <rdfs:subPropertyOf rdf:resource="#eventAccount"/>
```

```
</owl:ObjectProperty>
```

```
<!-- http://www.ipaw.info/2007/opm#triggeredByProcess -->
```

```
<owl:ObjectProperty rdf:about="#triggeredByProcess">
```

```
  <rdf:type rdf:resource="&owl;TransitiveProperty"/>
```

```
  <rdfs:comment rdf:datatype="&xsd:string"
```

```

        >the process that triggered the process</rdfs:comment>
<rdfs:range rdf:resource="#Process"/>
<rdfs:domain rdf:resource="#Triggered"/>
<rdfs:subPropertyOf rdf:resource="#eventProcess"/>
<owl:inverseOf rdf:resource="#triggeredProcess"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#triggeredProcess -->

<owl:ObjectProperty rdf:about="#triggeredProcess">
  <rdfs:type rdf:resource="#owl:TransitiveProperty"/>
  <rdfs:comment rdf:datatype="#xsd:string"
    >the process that was triggered</rdfs:comment>
  <rdfs:range rdf:resource="#Process"/>
  <rdfs:domain rdf:resource="#Triggered"/>
  <rdfs:subPropertyOf rdf:resource="#eventProcess"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#triggeredTime -->

<owl:ObjectProperty rdf:about="#triggeredTime">
  <rdfs:comment rdf:datatype="#xsd:string"
    >the time at which the process was triggered</rdfs:comment>
  <rdfs:domain rdf:resource="#Triggered"/>
  <rdfs:subPropertyOf rdf:resource="#eventTime"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#usedAccount -->

<owl:ObjectProperty rdf:about="#usedAccount">
  <rdfs:domain rdf:resource="#Used"/>
  <rdfs:subPropertyOf rdf:resource="#eventAccount"/>

```

```
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#usedArtifact -->

<owl:ObjectProperty rdf:about="#usedArtifact">
  <rdfs:comment rdf:datatype="&xsd:string"
    >the artifact that was used</rdfs:comment>
  <rdfs:range rdf:resource="#Artifact"/>
  <rdfs:domain rdf:resource="#Used"/>
  <rdfs:subPropertyOf rdf:resource="#eventArtifact"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#usedByProcess -->

<owl:ObjectProperty rdf:about="#usedByProcess">
  <rdfs:comment rdf:datatype="&xsd:string"
    >the process that used the artifact</rdfs:comment>
  <rdfs:range rdf:resource="#Process"/>
  <rdfs:domain rdf:resource="#Used"/>
  <rdfs:subPropertyOf rdf:resource="#eventProcess"/>
</owl:ObjectProperty>

<!-- http://www.ipaw.info/2007/opm#usedRole -->

<owl:ObjectProperty rdf:about="#usedRole">
  <rdfs:comment rdf:datatype="&xsd:string"
    >the role the artifact played as input into the process
    </rdfs:comment>
  <rdfs:range rdf:resource="#Role"/>
  <rdfs:domain rdf:resource="#Used"/>
  <rdfs:subPropertyOf rdf:resource="#eventRole"/>
</owl:ObjectProperty>
```

```

<!-- http://www.ipaw.info/2007/opm#usedTime -->

<owl:ObjectProperty rdf:about="#usedTime">
  <rdfs:comment rdf:datatype="&xsd:string"
    >the time at which the artifact was used</rdfs:comment>
  <rdfs:domain rdf:resource="#Used"/>
  <rdfs:subPropertyOf rdf:resource="#eventTime"/>
</owl:ObjectProperty>

<!--
////////////////////////////////////
//
// Data properties
//
////////////////////////////////////
-->

<!-- http://www.ipaw.info/2007/opm#name -->

<owl:DatatypeProperty rdf:about="#name">
  <rdfs:comment rdf:datatype="&xsd:string"
    >a name given to this OPM entity</rdfs:comment>
  <rdfs:range rdf:resource="&xsd:string"/>
  <rdfs:domain>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
        <rdf:Description rdf:about="#Account"/>
        <rdf:Description rdf:about="#Node"/>
        <rdf:Description rdf:about="#Role"/>
      </owl:unionOf>
    </owl:Class>
  </rdfs:domain>
</owl:DatatypeProperty>

```



```

    </rdfs:domain>
</owl:DatatypeProperty>

<!-- http://www.ipaw.info/2007/opm#noEarlier -->

<owl:DatatypeProperty rdf:about="#noEarlier">
  <rdfs:comment rdf:datatype="&xsd:string"
    >the event occurred no earlier than this time</rdfs:comment>
  <rdfs:domain rdf:resource="#TimeInterval"/>
  <rdfs:range rdf:resource="&xsd:dateTime"/>
</owl:DatatypeProperty>

<!-- http://www.ipaw.info/2007/opm#noLater -->

<owl:DatatypeProperty rdf:about="#noLater">
  <rdfs:comment rdf:datatype="&xsd:string"
    >the event occurred no later than this time</rdfs:comment>
  <rdfs:domain rdf:resource="#TimeInterval"/>
  <rdfs:range rdf:resource="&xsd:dateTime"/>
</owl:DatatypeProperty>

<!--
////////////////////////////////////
//
// Classes
//
////////////////////////////////////
-->

<!-- http://www.ipaw.info/2007/opm#Account -->

<owl:Class rdf:about="#Account">

```

```
        <rdfs:subClassOf rdf:resource="#OpmEntity"/>
</owl:Class>

<!-- http://www.ipaw.info/2007/opm#Agent -->

<owl:Class rdf:about="#Agent">
    <rdfs:subClassOf rdf:resource="#Node"/>
</owl:Class>

<!-- http://www.ipaw.info/2007/opm#Artifact -->

<owl:Class rdf:about="#Artifact">
    <rdfs:subClassOf rdf:resource="#Node"/>
</owl:Class>

<!-- http://www.ipaw.info/2007/opm#Controlled -->

<owl:Class rdf:about="#Controlled">
    <rdfs:subClassOf rdf:resource="#Event"/>
</owl:Class>

<!-- http://www.ipaw.info/2007/opm#Derived -->

<owl:Class rdf:about="#Derived">
    <rdfs:subClassOf rdf:resource="#Event"/>
</owl:Class>

<!-- http://www.ipaw.info/2007/opm#Event -->

<owl:Class rdf:about="#Event">
    <rdfs:subClassOf rdf:resource="#OpmEntity"/>
</owl:Class>
```

```
<!-- http://www.ipaw.info/2007/opm#Generated -->

<owl:Class rdf:about="#Generated">
  <rdfs:subClassOf rdf:resource="#Event"/>
</owl:Class>

<!-- http://www.ipaw.info/2007/opm#Node -->

<owl:Class rdf:about="#Node">
  <rdfs:subClassOf rdf:resource="#OpmEntity"/>
</owl:Class>

<!-- http://www.ipaw.info/2007/opm#OpmEntity -->

<owl:Class rdf:about="#OpmEntity"/>

<!-- http://www.ipaw.info/2007/opm#Process -->

<owl:Class rdf:about="#Process">
  <rdfs:subClassOf rdf:resource="#Node"/>
</owl:Class>

<!-- http://www.ipaw.info/2007/opm#Role -->

<owl:Class rdf:about="#Role">
  <rdfs:subClassOf rdf:resource="#OpmEntity"/>
</owl:Class>

<!-- http://www.ipaw.info/2007/opm#TimeInterval -->

<owl:Class rdf:about="#TimeInterval">
```

```
        <rdfs:subClassOf rdf:resource="#OpmEntity"/>
</owl:Class>

<!-- http://www.ipaw.info/2007/opm#Triggered -->

<owl:Class rdf:about="#Triggered">
    <rdfs:subClassOf rdf:resource="#Event"/>
</owl:Class>

    <!-- http://www.ipaw.info/2007/opm#Used -->

<owl:Class rdf:about="#Used">
    <rdfs:subClassOf rdf:resource="#Event"/>
</owl:Class>
</rdf:RDF>
```