Undefined 1 (2012) 1–5
IOS Press



# A Survey on Semantic Scientific Workflow

**Editor(s):** Name Surname, University, Country
**Solicited review(s):** Name Surname, University, Country
**Open review(s):** Name Surname, University, Country

Zhili Zhao, Adrian Paschke *
*Department of Mathematics and Computer Science, Free University Berlin, Berlin, Germany*
*E-mail: zhaozhil03@gmail.com, paschke@mi.fu-berlin.de*

**Abstract.** Over the last ten years, scientific workflows have become an important technology in modern scientific computation, which facilitates scientists to perform data management, analysis, and simulation, e.g., in scientific experiments. Flexible workflow design, adaptable execution models and reproducibility are the fundamental requirements of scientific workflows. In this paper, we identify several critical elements for flexible and adaptable scientific workflow management systems and provide an extensive survey of current efforts in the Semantic Web community to combine and use semantic technologies for scientific workflows.

Keywords: Scientific Workflow, Flexibility, Adaptation, Exception Handling, Reproducibility

## 1. Introduction

In recent years, scientific workflows are gaining more and more attention to support scientific computations and experimenting. A scientific workflow represents and manages complex distributed computations and accelerates the pace of scientific progress in many scientific areas [49], such as: astronomy, ecology, bioinformatics, earth science and etc. Scientists can benefit from an explicitly modeled and executed workflow not only because it utilizes various resources from different administrative domains and automates troublesome experimental process, but also automatically captures provenance information in detail, which is critical for further verification, analysis and new discovery.

Workflow technology was first adopted in the business domain for business processes to optimize an organization's processes in an administrative context. Over the years, many competing specifications and standards were proposed, some of which have become broadly accepted and used, superseding others [8]. For instance, the Business Process Execution Language (BPEL) [4] is a standard way of orchestrating Web service execution in a business domain and the Business Process Model and Notation (BPMN) has become a broadly accept way to model business processes. However, scientific workflows have some extra requirements over their business counterparts, and it is inadvisable to directly reuse the technologies from the business community.

In contrast to traditional business workflows, scientific workflows are exploratory in nature and often executed in a what-if or trial-and-error manner [45,6]. Their outcome might be used to confirm or invalidate a scientific hypothesis or serve some similar experimental goals, which involve many repetitive, synchronous, and concurrent tasks. Scientific workflows are more dataflow-oriented and data is often streamed through independent processes. Furthermore, scientific workflows are usually executed in an evolving environment, where distributed resources integrated are not only heterogeneous, but also may come and disappear at any time. Therefore, a scientific workflow system which is resilient to the volatile execution environment and sup-

*Corresponding author. E-mail: paschke@mi.fu-berlin.de.

ports dynamic and adaptive workflow execution is the ultimate goal of the scientific workflow community. To achieve this, firstly, it is necessary to provide a flexible scientific workflow specification, which supports adaptive execution based on the real-time context during execution. Besides, the specification should provide an abstraction level on different heterogeneous resources and facilitate workflow engines to select available concrete resources at runtime. This goal also imposes some new challenges, such as: methodologies are required to support the mapping between the abstract description and concrete resources and to provide flexible exception handling strategies; and it is more complicated to record provenance information in such a highly dynamic and heterogeneous environment.

Over the years, different solutions to these challenges have been proposed, in order to improve the flexibility and adaptability of scientific workflows. Although, there are numerous independent efforts to make scientific workflows more flexible and adaptable, there is a new tendency towards the inclusion of semantic information, ontologies, and execution rules inside the workflow execution [2]. Currently, the Semantic Web is a very active community engaging in incorporating semantics into traditional resources and many innovative technologies and standards have been proposed. The Semantic Web extends traditional web resources with additional metadata and semantic knowledge and allows knowledge to be shared and reused across applications, enterprises, and community boundaries. It is built on W3C's Resource Description Framework[1] (RDF, which is a metadata data model for data interchange on the Web), and Web Ontology Language[2] (OWL, which explicitly represents the relationships between things based on RDF) and the Rule Interchange Format (RIF) and RuleML[3]. The inclusion of Semantic Web technologies in scientific workflow gives many advantages. Incorporating semantics to workflow specification provides a more natural and powerful language, which enables scientists to capture scientific process in a flexible and abstract way. By adding semantic annotations into Web Services, which are regarded as a unit of workflow for completing certain goals, it enables workflow engines to discover and select the optimal services at runtime. Moreover, the inclusion of semantic information into provenance data allows applications unambiguously interpret data in the correct context. It is worth noticing that we call the scientific workflow, which supports flexible design and adaptable execution based on Semantic Web technologies, a semantic scientific workflow in the survey.

Existing semantic related efforts in scientific workflow systems have not been systematically studies yet, which led to broad variety of solutions, ranging from adding semantic information to existing Web Services to rule-based execution at runtime. In this survey, we review existing efforts which incorporate the achievements of Semantic Web community into scientific workflows for the purpose of making them more flexible and adaptable. Based on this survey we will make some suggestions for the future development of scientific workflows.

The rest of the paper is structured as follows: Section 2 describes some key requirements of scientific workflows. Based on them, we identify several critical elements to a flexible and adaptable scientific workflow: flexible scientific workflow definition, semantic service description, adaptive workflow execution and semantic provenance. Section 3, 4, 5 and 6 introduces existing efforts from the Semantic Web community on each element and provides critical analysis in terms of the requirements of scientific workflow. Finally, Section 7 makes a number of suggestions for the future development of scientific workflow and concludes the paper.

## 2. Scientific Workflow Requirements Analysis

Compared to traditional business workflows, scientific workflows have an extra set of requirements, which is the reason why many efforts designed and developed scientific workflow systems from the scratch, instead of reusing mature specifications from business domain. In what follows, we firstly present some key requirements of scientific workflows. Based on them, we identify several important elements for flexible scientific workflows and analyse how they benefit from Semantic Web technologies.

### *2.1. Key Requirements of Scientific Workflow*

Typical features of scientific workflows and their challenges have been identified based on a comparative study with traditional workflow in business do-

[1] http://www.w3.org/RDF/
[2] http://www.w3.org/TR/owl-features/
[3] http://www.w3.org/2005/rules/wiki/RIF_Working_Group

main [31,49,44,8,48,6]. Here, we enumerate some of the relevant requirements.

**Service composition & reuse** A workflow is the composition of services and their dependencies to complete a more complex work. It is not only necessary to reuse single services, but also to treat a workflow itself as a senior service and incorporate into another workflow. In other words, both of them need to be represented and managed in uniform way.

**Flexibility** denotes the ability of a workflow system to react to changes in its environment. Scientific workflows need a flexible design to support exploration and a flexible modification based on dynamic context at runtime.

**Scalability** With the experimental goal, scientific workflows are usually executed in an exploratory way and might require resources that are not predefined. It should scale with number of utilized services, data or calculation resources.

**High Usability** Scientists usually are non-computer experts, and a scientific workflow system should hide the complexity of underlying infrastructures and allow the same information to be shown at various levels of abstraction, depending on who is using the system.

**Reliability and fault-tolerance** denotes the ability of being failure-resistant, since scientific workflows are often executed in an evolving environment with heterogeneous resources.

**Reproducibility** A scientific workflow should be reproducible and record the specific details of creating a derived data product. The provenance data logs the sequence of steps, parameter settings and intermediate products and are very helpful for scientists to repeat a workflow or to validate their assumptions.

### *2.2. Important Elements to Semantic Scientific Workflow*

The requirements mentioned above portray the different features of scientific workflow, however, they are not independent to each other. This is the reason why scientific workflows have not been widespread applied as their business workflow counterpart.

The complexity of a scientific workflow lies in its experimental goal and unreliable execution environment. Scientific workflows are usually executed in a highly heterogeneous and distributed environment, which is unreliable and evolving all over the time. For the purpose of improving reliability and handling various exceptions, an adaptive execution which dynamically selects available services or modifies the structure of a workflow at runtime runs first in the development of a scientific workflow.

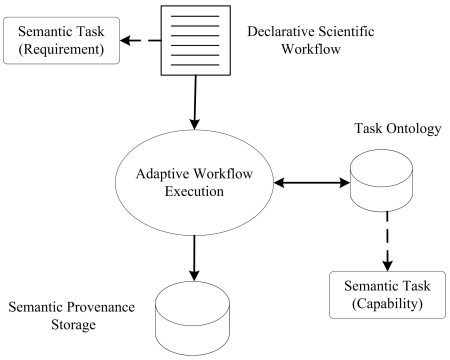


Fig. 1. Rule Responder Interface Description

Adaptive execution not only involves concrete service selection and exceptions handling in a workflow engine that should have knowledge to support this, but also the workflow specification should be flexible and abstract to accommodate service discovery at runtime. To achieve this, it is promising to incorporate semantics into both workflow specification and service description. The semantic description of services generally includes its syntactic structure and validation rules (e.g. precondition, postcondition, etc.), which are used for discovery and evaluation of resources at runtime. On the other hand, a workflow specification enhanced with semantic rules facilitates scientists to describe more complex decision and behavioral logics and also accommodates adaptable execution with more intelligent strategies, such as: following alternative execution paths in case of errors or unexpected exceptions.

Another important element of scientific workflow is provenance, which is used for verification, reproduction and analysis of scientific workflows. It becomes even more important in the context of flexible and dynamic execution environments. And, with the exponentially increasing volumes of data from scientific experiments, it is necessary to employ semantics into provenance in order to unambiguously interpret data in the correct context.

Of course, there are other important elements of scientific workflows, such as: workflow scheduling, data movement, etc. We can see that flexible workflow composition, adaptive workflow execution, semantic task description and semantic provenance are the most significant elements dominating the flexible and adaptive of scientific workflow, as shown in Figure 1. Each element plays a critical role for the development of flexible and adaptable scientific workflows in future. In

what follows, we look at each element and analyse existing proposals in detail.

## 3. Flexible Scientific Workflow Definition

Flexible scientific workflows require a flexible design of complex experimental logics, but also an intuitive way to describe user requirements to find a concrete service at runtime. In this section, we present some approaches for flexible workflow design and assess them.

### *3.1. OWL-S/OWL-WS based Workflow Definition*

OWL-S[4] builds on OWL and describes the properties and features of web service in a machine-readable markup language (We will present how it describes Web Services in Section 4.1). Besides, OWL-S also provides a composite process to express most of basic control and data flows constructs necessary for specifying a workflow of services, such as: sequence, if-then-else, split + join, choice, condition, iterate, etc. However, OWL-S focuses on modeling a workflow that is internal to a single service, i.e. an OWL composite process (workflow) specifies the steps interacting with a single service implementation, which does not accord with the reality. OWL-WS [11] is an extended version of OWL-S invented by NextGrid[5] project and standards for OWL for Workflows and Services. It enforces OWL-S and allows a grounding being composed by components referring different services to describe more realistic and complicated processes. However, the application of both OWL-S and OWL-WS hasn't been prevalent for lack of tools to support the development [40]. In addition, there is no mechanism designed for handling exceptions at runtime.

### *3.2. Rule-based Service Composition*

Rule-based approaches are another way to support more flexible service composition and model the logic of process with a set of rules using declarative languages. In [2], Marc Frincu et al. look at scientific workflows from a distributed system perspective and shot that rule-based workflow composition has advantages to handle issues related to scalability, failure tolerance, data integrity and scheduling. They give an overview on typical workflow issues and solutions and present a simple ECA rule-based workflow formalism for self-adaptation and auto-generation. [18] also describes an ECA-based workflow management system for service composition. An automatic event composition algorithm is developed to automate the event processing and validate the manual activities composition at design time. Compared to [2], [18] and its evolutionary work [46] are more sophisticated and a prototype is given. [47] proposes a declarative and rule-driven framework to dynamic service composition, while its ramification are further explored and illustrated with a realistic case study. In contrast to other efforts in rule-based workflow formalism, they also aim at providing flexible workflow orchestration in the business domain [14,30,40]. We can see that, the work mentioned above primarily focus on a flexible design, and are limited to other important requirements of scientific workflows, such as: exception handling, reproducibility, etc. Moreover, a rule-based workflow definition is usually very complicated for non-IT scientists, who have to learn extra knowledge before using it.

### *3.3. Agent-based Service Composition*

Adam Barker et al. [10,9] propose to capture scientific processes with the MultiAgent Protocol (MAP), which allows the typical features of scientific workflow requirements to be understood in terms of pure coordination and to be executed in an agent-based, decentralized, peer-to-peer architecture. The authors present a motivating scientific workflow taken from the Large Synoptic Survey Telescope (LSST) and show how the agent-based approach is helpful to classify previously unknown objects. Each agent taking part in the interaction adopts a role, by which the agent references a reasoning Web Service that implements all the decision procedures required for that role type. [16] presents an approach to specify a workflow as multi-agent system, which can intelligently adapt to changing environmental condition. The authors of [16] argue that Adaptive Workflow = Web Services + Agents, where the Web services provide computational resources and the Agents provides a coordination framework. The initial social order of a multi-agent system is described by BPEL for Web Services (BPEL4WS). [28] describes an approach to build a multi-agent system, which can enact a set of workflows and cope with exceptions. The authors of [28] also represent how Semantic Web languages (such as: OWL-DL [24], SWRL [25]) can be used to describe organizational

[4] http://www.w3.org/Submission/OWL-S/
[5] http://www.nextgrid.org/

knowledge and domain knowledge and the agents can use this knowledge to make intelligent decisions at runtime.

A multi-agent system provides high scalability and can help to scale via distributed process execution. Besides, it is possible to provide extra reasoning intelligence inside an existing resource, which can be invoked by an agent as decision procedure. For example, adding policies into an agent to handle exceptions during invoking an existing service. However, the multi-agent system executes in a decentralized model, which provides high scalability but loses the strengths of centralized workflow execution, where the coordination of component process is centrally managed by a known coordinator and it is possible that alternative scenarios can be put in place in case faults occur [34].

### *3.4. Aspect-oriented Workflow Language*

The aspect-oriented approach is a paradigm for concern-based decomposition and aims to increase the modularity of a system. The key concepts of aspect-oriented programming language [26] are: join point, pointcut and advice. A join point is a point in the execution of a program and a point cut is one or more related join points span different processes. The advice is an activity that implements some crosscutting concern and is executed when a join point in the set identified by a pointcut is reached. Aspect activities are defined separately from process activities and provide a cross-process view on how a certain concern is handled in several workflows. AO4BPEL [17] is an aspect-oriented extention to BPEL, and modularizes various concerns such as measurement of activity time, auditing data collection, security, etc. However, since it is based on the BPEL, which not only cannot capture dynamic changes, but also offers a number of complex advanced features in business workflow, it is still not flexible enough for scientists to capture scientific processes.

### *3.5. Summary*

Although numerous solutions have been proposed, more and more efforts reach a consensus that scientific workflows require a declarative and flexible design to support exploration with Semantic Web technologies. Based on the literatures reviewed above, it seems the rule-based solution is the most suited for scientific processes description and provides many advantages over other approaches. However, the rule-based programming is usually complex, and a standard specification to describe services and their dependencies is essential. On the other hand, the agent-based approach has demonstrated its powerful strength compared to centralized workflow execution in the experiments requiring collaboration. The inclusion with semantic rules and facts into agents not only makes them more intelligent to make decision at runtime, but also enables them to request other agents to deal with unpredictable problems. Therefore, it would be promising to combine the rule-based language and the multi-agent system to capture the scientific processes. If so, the coordination of whole workflow process would be centrally managed by a known rule engine, which is also known as orchestration; a sub-process of workflow may be executed by multi-agent system in a choreography way which is explorable and collaborative.

A similar architecture has been proposed by [22]. For the purpose of improving the scalability of scientific workflow, the authors of [22] argue to model control flow within dataflow by combining orchestration and choreography. In other words, the overall workflow is modeled by orchestration that integrates sub-workflow, which on their part are modeled by choreography from a dataflow perspective. However, the service orchestration in their work is based on BPEL, which is not as flexible as rule languages. [13,39] introduces Rule Responder[6], which is a framework for specifying virtual organizations as semantic multi-agent to support collaborative teams. Human members of an organization are assisted by autonomous rule-based agents, which use Semantic Web rules to describe aspects of their owner's derivation and reaction logic. The solution provides a flexible and scalable framework to complete complex tasks. However, there is still much to be done when it comes to scientific workflow.

## 4. Semantic Service Description

In order to achieve an adaptable execution of scientific workflows, the specification of semantic scientific workflows requires being declarative and flexible. Besides the constraints of control flow, scientists also have to describe the constraints of involved task to enable a workflow engine to discovery concrete services at runtime. It is also necessary to describe existing ser-

[6]http://ruleml.org/RuleResponder/

vices with semantics for discovery, selection, invocation and composition.

In the following, we will use the term task to represent the abstract semantic information of a service to avoid confusion with its concrete instances with the execution information of the service. The semantic description includes not only syntactic structure, which reveals us what type of inputs/outputs it expects to receive, but also semantic information, which describes more complex validation rules for both inputs and outputs [2]. In this section, we present some prominent solutions of adding semantics into Web Services, and analyse their strength and weaknesses.

### *4.1. OWL-S*

Besides its composite process mentioned in Section 3.1, OWL-S enables users and software agents to automatically discover, invoke, compose, and monitor Web resources offering services, under specified constraints. OWL-S provides three essential types of knowledge about a service: ServiceProfile, ServiceModel and ServiceGroundling. The ServiceProfile and ServiceModel are abstract representations of a Web Service, while ServiceGrounding deals with the mapping from an abstract to a concrete specification of the service description, the most commonly used being Web Service Definition Language (WSDL). ServiceProfile provides a high-level description of the service, including functional attributes (e.g. input, output, precondition, results, etc.) and non-functional attributes (e.g. security, QoS, category, etc.), which can be used for service discovery. The work OWL-S was the first specification submitted to W3C in 2004 and has deeply affected the development of semantic web service. However, it hasn't been widespread applied because of its complexity and the top-down approach to modeling of services, which does not fit well with industrial developments of Service-Oriented Architecture (SOA).

### *4.2. WSDL-S*

WSDL-S[7] was originally proposed by the LSDIS laboratory at the University of Georgia and defines a mechanism to semantically annotate WSDL documents. Because of the weakness of top-down approach adopted by OWL-S, by extending the industry standards WSDL with extra elements and attributes, WSDL-S adds semantic information to represent the syntactic structure of a service. This approach is also known as bottom-up modeling of service. Semantic annotations are not tied to any particular ontology representation language and can be provided with different languages, such as: OWL, UML, etc. Compared to OWL-S, WSDL-S meets the practical situation better and can be easier applied.

### *4.3. SAWSDL*

Based on WSDL-S, the Semantic Annotations for WSDL and XML Schema (SAWSDL)[8] defines mechanisms using which semantic annotations can be added to WSDL components. SAWSDL does not specify a language for representing semantic models. Instead, it provides mechanisms by which concepts from the semantic models that are defined either within or outside the WSDL document can be referenced from within WSDL components as annotations. These semantics when expressed in formal languages can help disambiguate the description of Web services during automatic discovery and composition of the Web services. Similar with WSDL-S, it also adopts the bottom-up approach, but it is more open and doesn't prescribe a semantic framework to express incremental semantics.

### *4.4. WSMO*

The Web Service Modeling Ontology (WSMO)[9] provides means to describe all relevant aspects of Semantic Web services with the top-bottom approach. Taking the Web Service Modeling Framework (WSMF) [21] as a starting point, WSMO reuses its four different main elements for describing semantic Web Services: ontologies that provide the terminology used by other WSMO elements, Web service descriptions that define the functional and behavioral aspects of a Web service, goals that represent user desires, and mediators which handles interoperability problems between different WSMO elements. However, beside the weakness of the top-bottom approach, it provides several sophisticated mediators and has lower usability compared with other solutions.

[7] http://www.w3.org/Submission/WSDL-S/

[8] http://www.w3.org/2002/ws/sawsdl/

[9] http://www.w3.org/Submission/WSMO/

### 4.5. SWSF

Semantic Web Services Framework (SWSF)[10], which includes the Semantic Web Services Language (SWSL)[11] and the Semantic Web Services Ontology (SWSO)[12]. SWSL is used to specify formal characterizations of Web service concepts and descriptions of individual services. SWSO presents a conceptual model by which Web services can be described, and an axiomatization, or formal characterization, of that model. In contrast to WSMO, it is more focus on extending the functionality of the rule language.

### 4.6. WSMO-Lite

WSMO-Lite[13] is a lightweight set of semantic service descriptions proposed in 2007. It identifies the types and a simple vocabulary for semantic descriptions of services and fills the SAWSDL annotations with concrete semantic service descriptions. A conceptual model for Web Service Descriptions includes: Information Model Descriptions define the data model for input, output, and fault messages, as well as for the data relevant to the other aspects of the service description; Functional Descriptions define service functionality, nonfunctional descriptions, such as: price, QoS; Behavioral Descriptions define external and internal behavior, and Technical Descriptions define messaging details such as message serializations, communication protocols, and physical service access points.

### 4.7. Summary

Currently, all of specifications mentioned above have been submitted to W3C and have become member submissions except SAWSDL, which now is a W3C recommendation. However, all of them only provides a prototype and haven't been widespread applied. We can see that they either adopt the top-down or the down-up approach to add additional semantic information. The down-up approach builds increments on top of existing services and fits well with industrial developments of SOA technology. The top-down approach, on the other hand, seems more complex, but provides more comprehensive semantic information than its counterpart. Therefore, it is hard to say which approach is more superior. But, in future, there is still a lot to be done to make the process of incorporating semantics into services as simple as possible and provide more flexibility to users.

[10]http://www.w3.org/Submission/SWSF/
[11]http://www.w3.org/Submission/SWSF-SWSL/
[12]http://www.w3.org/Submission/SWSF-SWSO/
[13]http://www.w3.org/Submission/WSMO-Lite/

## 5. Adaptive Workflow Execution

During runtime, scientific workflows are executed in terms of both the specific context and the abstract workflow specification given by scientists. Firstly, it must be possible to bind abstract tasks to concrete services or even modify the structure of a workflow dynamically. Besides, since scientific workflow is executed in a dynamic and evolving environment with heterogeneous resources, exception failures are inevitable and may be caused by different reasons. In this section, we firstly describe a general process of task binding based on semantic description of workflow and services, and then present some classical approaches of exception handling at runtime.

### 5.1. Binding Abstract Tasks to Concrete Services

Resolving abstract tasks and binding them to concrete ones is fundamental in an adaptive execution of scientific workflow at runtime. As shown in Figure 2, the NextGRID project introduces a workflow-based application resolution, which involves three subprocesses: discovering candidate services, selecting the most suited candidate, and using the selected candidate. Discovering candidate services depends on the description of an abstract task (i.e. semantic constraints) and service registry (which assembles services with both semantic and execution information). The result of discovery is a list of candidate services, which are capable of completing the task (i.e. all candidates have the same effect). Selecting candidate includes evaluating the capability of candidate services and getting the most suited one based on the service requirements of a task given by scientists. Usually, there are many different criteria used to evaluate candidate services, such as: the least estimated execution time, the least-expensive and etc. Before using the selected candidate, the abstract task will be replaced and some operations are needed to encapsulate the abstract task into the selected candidate service, such as: parameters assignment, etc.

Additionally, we can see in Figure 2 that, each subprocess is implemented as an independent workflow and incorporated into the workflow at runtime, allow-

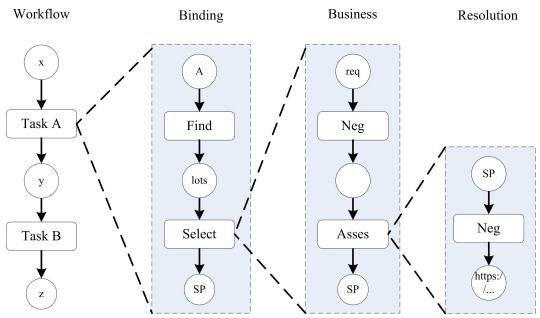


Fig. 2. Workflow-based Application Resolution [5]

ing abstract workflow to be resolved to specific services at specific endpoints. In other words, each workflow (sub-process) is presented and treated as other single tasks in uniform way. Moreover, for the purpose of improving usability, they are usually implemented as the core functionalities of scientific workflow systems and leave scientists focus on their experimental process.

### 5.2. Exception Handling

Many different strategies have been proposed to handle the exceptions of scientific workflows at runtime, ranging from some simple policies (such as retry, checkpoint/restart [36], replication [3]) to sophisticated exception handling involving users. Here, we introduce some general exception handling strategies and try to find out how to support these strategies in semantic scientific workflow.

#### 5.2.1. Exception Propagation

The exception propagation is mostly used in the hierarchical workflows and has been studied in many efforts [45,20,23,33]. With this pattern, an exception hander can propagate an exception to a more appropriate higher level if the lower level cannot handle it. Generally speaking, with the perspective of workflow composition, scientific workflow system can be divided into four levels: workflow level, task level, resource level and system level. At workflow level, is also known as application level or structure level, where a workflow is described by tasks and their dependencies, task level concerns each individual task of the workflow, resources level concerns about the resources required by the workflow, system level refers to concrete execution environment. Each layer usually corresponds to different types of exceptions. For instance, on the resource layer a file is lost or network was unreachable in system level. If an exception is propagated to the root without getting caught by an exception handler, other specialized exception handling mechanisms can be employed to deal with this situation.

#### 5.2.2. Dynamic Replacement

Similar to the abstract task binding, dynamic replacement refers to treatments of an exception by dynamically replacing an exceptional service with an alternative owning the same effect [45,28]. In semantic scientific workflows, this could be seen as rebinding an abstract task to another available service. The predecessor and successor of exceptional process knows nothing about the replacement. Of course, there would be a situation, where no candidate is available. In this case other exception handling strategies might be employed, such as exception propagation or asking users for help.

#### 5.2.3. Human Steering Exception Handling

Although the vision of scientific workflows aims to automate scientific process, scientists are still required to conduct some manual tasks or make complicated decisions at runtime. Some efforts have been made to support human steering in workflow, such as BPEL4People[14] and WS-HumanTask[15]. Both of them make it possible to wrap human behaviour into Web Service and enable the integration of human beings in SOA systems. In Triana [19], fault tolerance is generally user driven. In case of an exceptions singling, a dialog with the exception and its related context information is popped up and asks scientists to make decision. As to other fault-tolerant solutions supporting user intervention, most of them aim to involve users to handle unexpected exceptions [37,38]. It seems that these approaches violate the original vision of scientific workflow after involving scientists into the execution of scientific workflow, however, scientists could provide additional knowledge and it really benefits to handle some complicated exceptions.

#### 5.2.4. Knowledge-based Exception Handling

In a knowledge-based approach the exception handling strategy either reuses the stored experience to handle exceptions, or finds an alternative execution path via reasoning semantic ontologies. The solution in [27] provides a knowledge base concerning what

[14] http://www.oasis-open.org/committees/bpel4people/
[15] http://docs.oasis-open.org/bpel4people/ws-humantask-1.1.html

kinds of exceptions can occur in collaborative work processes, and how these exceptions can be handled. When an exception occurs, enactment-time tools are provided to help diagnose their underlying causes and suggest specific interventions to handle it. [32] proposes to reuse exception handing experiences, which is also known as Case-Based Reasoning (CBR). In other words, an exception handler analyses a case repository and finds similar experience to handle exceptions. [29] represents a decentralised multi-agent system to deal with unexpected exceptions. Each agent are endowed with semantic knowledge (in OWL) about the capabilities and relationships with other agents so that they can deal with exceptions via reasoning ontologies.

### *5.3. Summary*

Dynamic task binding and exception handling are two important parts of adaptive scientific workflow execution. To support them, there are several requirements which need to be considered. First, both of them should be modularized. Dynamic task binding and exception handling are needed in many situations and cut across workflow process. The modularity of them can benefit the understanding and reuse of scientific workflows. Second, both of them should be separated from normal processes and provided by a scientific workflow system to make the normal processes as simple as possible. Third, semantic knowledge is critical for dynamic task binding and exception handling at runtime. Therefore, it is very helpful to provide a flexible workflow specification and incorporate semantics into heterogeneous services. For instance, in order to select the most suited services, QoS related properties should be incorporated into semantic service description. Fourth, besides the integration with with the involvement of scientists, it is also necessary to combine different handling strategies together to provide a sophisticated solution.

Last but not the least, since the dynamic runtime changes at runtime, event-based exception handling seems to be a promising one. Exceptions arising at runtime can be detected as events and handlers reacts them based on its rules and knowledge base.

## 6. Semantic Provenance

Metadata and Provenance are critical to effectively manage the exponentially increasing volumes of scientific data from scientific experiments. However, traditional efforts of scientific workflow provenance domain concentrate on a "workflow engine perspective of the world" [41], i.e. the provenance information collected is only the detailed traces of data transformation, such as: operations along with input and out files, etc. Theses traces are a form of metadata, relative to the data involved in the process, known as data provenance. With the challenge of increasing scientific data and the development of Semantic Web technologies, it is necessary to incorporate semantics into provenance information. This process is also known as semantic provenance which, based on domain-specific provenance ontologies, lets software applications unambiguously interpret data in the correct context [41].

### *6.1. Open Provenance Model*

The Open Provenance Model (OPM)[16] was the result of the Provenance Challenge series that was initiated in May 2006. Because of the heterogeneity of numerous provenance systems, OPM provides a generic provenance model to improve interoperability of different provenance models. Based on the three primary entities: Agent (Contextual entity acting as a catalyst of a process, enabling, facilitating, controlling, affecting its execution), Artifact (Immutable piece of state, which may have a physical embodiment in a physical object, or a digital representation in a computer system) and Process (Action or series of actions performed on or caused by artifacts, and resulting in new artifacts), and the causal dependencies between them, the provenance model is represented by a directed graph. OPM is an abstract model, both Open Provenance Model Vocabulary (OPMV)[17] and Open Provenance Model Vocabulary (OPMO)[18] implements it with different expressivity and reasoning. OPMV is a lightweight provenance vocabulary and can be used together with other provenance-related RDF/OWL vocabularies/ontologies, such as Dublin Core, FOAF, the Changeset Vocabulary, and the Provenance Vocabulary. On the other hand, OPMO uses more complex OWL 2.0 constructs to define more constraints and supports full expressivity and reasoning.

Since OPM was devised, many existing scientific workflow systems enhanced their provenance subsystem to support OPM [15,1,7,43]. Moreover, [42] summarizes four different approaches to harmonize

[16] http://openprovenance.org/

[17] http://open-biomed.sourceforge.net/opmv/ns.html

[18] http://openprovenance.org/model/opmo

the existing provenance systems with OPM: tightly coupled integration, loosely coupled storage integration, loosely coupled query integration and fully decoupled. The authors of [42] identify and analyse the relative merits of these approaches and help existing developers to determine the most suited one for them. However, most of these efforts focus on capturing the detailed traces of workflow, which have limit connection with external semantic resources.

### *6.2. Semantic Provenance as Linked Open Data*

For the purpose of effectively enabling software agents not only to "compute" over provenance information, but also to use it to accurately interpret eScience data in the correct context, Sytaya S. Sahoo et al. argue that incorporating domain knowledge and ontological underpinning in provenance using expression domain-specific provenance ontologies [41]. Based on the Component-Based Software Engineering principle and on the development in service-oriented computing (SOC), they proposed a approach of "two degrees of separation" in order to decouples the task of generating high-quality semantic provenance from the core functionality of workflow engines. The semantic provenance generation task is managed by specialized services that refer to domain-specific provenance ontologies.

In their evolutionary work [35], they take a concrete step towards the implementation of a semantic provenance model, called Janus, is used to record semantic provenance information in life science workflows. Additionally, the authors of [35] present the connecting of domain-enhanced provenance graphs with the global Web of Data, which is uniformly represented according to the principles of Linked Open Data (LOD) [12] to expand possible semantic province queries.

### *6.3. Summary*

Over past years, many efforts have been made to support recording provenance information generated during the execution of scientific workflows. In order to improve the interoperability of heterogeneous provenance systems, the OPM is proposed to represent provenance information in 2006. However, there is still much to be done. With the complexity of data management increasing dramatically, it is essential not only to disambiguate data and enable reuse, but also to incorporates semantics into scientific workflow provenance and enables the use of reasoning tools to perform deeper analysis. Furthermore, since different disciplines require different kinds of grain of provenance information, it is impossible to capture all the necessary details in each experiments. Besides the general provenance information, how to collect the customized provenance information and link them with external semantic resources is required.

## 7. Conclusion and Discussion

In this survey, we introduced several key requirements of scientific workflows and identified flexible workflow design, semantic service description, adaptive workflow execution and reproducibility as the critical elements for semantic scientific workflows. We can see that, all of them aim at making scientific workflows to be executable in evolving distributed heterogeneous environments. We surveyed several solutions which employ the technologies from the Semantic Web community.

Regarding flexible workflow design, a combination of rule-language-based orchestration and multi-agent-based choreography seems to be a promising solution. With the declarative programming approach, a rule-based language allows scientists to describe what the outcome should be, rather than specifying how to do it. Additionally, it could be executed by a centralized rule engine that controls the whole process of the workflow. Rule-based agents can be used to perform one or more tasks. Such agents can utilize their internal intelligence (i.e. rules and facts) to find the most suited service to complete the task or even handle failures at runtime. An agent could dynamically request other agents to be involved in workflow and form a scalable collaborative environment.

When it comes to execution, rule-based workflows, especially ECA-based workflows, meet the actual execution environment of scientific workflows better and give lots of advantages. First, it makes possible to modularize the crosscutting concerns and separates them from normal processes. Second, it is also a better choice to detect dynamic runtime changes and facilitates exception handling.

Rule-based workflow execution also provides a convenient way to follow the execution of workflow and records it in detail. But also, it facilitates external semantic ontologies to be incorporated into provenance information.

In summary, it has become a tendency to employ declarative rules, semantic-based description,

knowledge-based agent systems, ontologies and other technologies from Semantic Web Community into scientific workflow. However, there is still which needs to be done. For instance, rule-based languages are usually very complex for non-computer scientists and it is necessary to hide this complexity and improve the usability of rule-based scientific workflows. Besides, standards and frameworks for workflow specification and execution are still missing.

## References


[1] Data Lineage Model for Taverna Workflows with Lightweight Annotation Requirements. pages 17–30. 2008.

[2] *Dynamic and Adaptive Rule-Based Workflow Engine for Scientific Problems in Distributed Environments*, chapter 10, pages 227–251. CRC Press, 2010.

[3] J. H. Abawajy. Fault-tolerant scheduling policy for grid computing systems. *Parallel and Distributed Processing Symposium, International*, 14:238b, 2004.

[4] Tony Andrews, Francisco Curbera, Hitesh Dholakia, Yaron Goland, Johannes Klein, Frank Leymann, Kevin Liu, Dieter Roller, Doug Smith, Satish Thatte, Ivana Trickovic, and Sanjiva Weerawarana. Business process execution language for web services version 1.1. Technical report, BEA, IBM, Microsoft, SAP, Siebel, 2003.

[5] Nikolaos Matskanis Mike Surridge Fabrizio Silvestri Barbara Cantalupo, Ludovico Giammarino. Semantic workflow representation and samples. Technical report, University of Southampton IT Innovation Centre, 2005.

[6] Roger Barga and Dennis Gannon. Scientific versus business workflows. In Ian J. Taylor, Ewa Deelman, Dennis B. Gannon, and Matthew Shields, editors, *Workflows for e-Science*, pages 9–16. Springer London, 2007.

[7] Roger S. Barga, Yogesh L. Simmhan, Eran Chinthaka, Satya Sanket Sahoo, Jared Jackson, and Nelson Araujo. Provenance for scientific workflows towards reproducible research. *IEEE Data Eng. Bull.*, 33(3):50–58, 2010.

[8] Adam Barker and Jano Hemert. Scientific workflow: A survey and research directions. In Roman Wyrzykowski, Jack Dongarra, Konrad Karczewski, and Jerzy Wasniewski, editors, *Parallel Processing and Applied Mathematics*, volume 4967 of *Lecture Notes in Computer Science*, pages 746–753. Springer Berlin Heidelberg, 2008.

[9] Adam Barker and Robert G. Mann. Agent-based scientific workflow composition. volume 351, pages 485–488, 2006.

[10] Adam Barker and Robert G. Mann. Flexible service composition. In *CIA*, pages 446–460, 2006.

[11] S. Beco, B. Cantalupo, L. Giammarino, N. Matskanis, and M. Surridge. Owl-ws: a workflow ontology for dynamic grid service composition. In *e-Science and Grid Computing, 2005. First International Conference on*, pages 8 pp. –155, july 2005.

[12] C. Bizer, T. Heath, and T. Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1âĂŞ22, 2009.

[13] Harold Boley and Adrian Paschke. Rule responder agents framework and instantiations. In Atilla ElÃği, Mamadou KonÃl', and Mehmet Orgun, editors, *Semantic Agent Systems*, volume 344 of *Studies in Computational Intelligence*, pages 3–23. Springer Berlin / Heidelberg, 2011.

[14] Mohamed Boukhebouze, Youssef Amghar, Aïcha-Nabila Benharkat, and Zakaria Maamar. A rule-based modeling for the description of flexible and self-healing business processes. In *Proceedings of the 13th East European Conference on Advances in Databases and Information Systems*, ADBIS '09, pages 15–27, Berlin, Heidelberg, 2009. Springer-Verlag.

[15] Shawn Bowers, Timothy M. McPhillips, Sean Riddle, Manish Kumar Anand, and Bertram Ludäscher. Kepler/ppod: Scientific workflow and provenance support for assembling the tree of life. In *IPAW*, pages 70–77, 2008.

[16] Paul A. Buhler and Jose M. Vidal. Adaptive workflow = web services + agents. In *In Proceedings of the International Conference on Web Services*, pages 131–137. CSREA Press, 2003.

[17] Anis Charfi and Mira Mezini. Ao4bpel: An aspect-oriented extension to bpel. *World Wide Web*, 10:309–344, 2007. 10.1007/s11280-006-0016-3.

[18] Lin Chen, Minglu Li, and Jian Cao. A rule-based workflow approach for service composition. In *Proceedings of the Third international conference on Parallel and Distributed Processing and Applications*, ISPA'05, pages 1036–1046, Berlin, Heidelberg, 2005. Springer-Verlag.

[19] David Churches, Gabor Gombas, Andrew Harrison, Jason Maassen, Craig Robinson, Matthew Shields, Ian Taylor, and Ian Wang. Programming scientific and distributed workflow with triana services: Research articles. *Concurr. Comput. : Pract. Exper.*, 18(10):1021–1037, August 2006.

[20] Xubo Fei and Shiyong Lu. A dataflow-based scientific workflow composition framework. *IEEE Transactions on Services Computing*, 5:45–58, 2012.

[21] Dieter Fensel and C. Bussler. The web service modeling framework wsmf. pages 17–20, 2002.

[22] Tino Fleuren, Joachim Gotze, and Paul Muller. Workflow skeletons: Increasing scalability of scientific workflows by combining orchestration and choreography. *Web Services, European Conference on*, 0:99–106, 2011.

[23] Minmin Han, Thomas Thiery, and Xiping Song. Managing exceptions in the medical workflow systems. In *Proceedings of the 28th international conference on Software engineering*, ICSE '06, pages 741–750, New York, NY, USA, 2006. ACM.

[24] Ian Horrocks and Peter F. Patel-Schneider. Reducing owl entailment to description logic satisfiability. In *Journal of Web Semantics*, pages 17–29. Springer, 2003.

[25] Ian Horrocks, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Technical report, World Wide Web Consortium, May 2004.

[26] Gregor Kiczales and Erik Hilsdale. Aspect-oriented programming. *SIGSOFT Softw. Eng. Notes*, 26(5):313–, September 2001.

[27] Mark Klein and Chrysanthos Dellarocas. A knowledge-based approach to handling exceptions inworkflow systems. *Comput. Supported Coop. Work*, 9(3-4):399–412, August 2000.

[28] Joey Lam, Frank Guerin, Wamberto Vasconcelos, and Timothy J. Norman. Building multi-agent systems for workflow enactment and exception handling. In *Proceedings of the 5th international conference on Coordination, organizations, institutions, and norms in agent systems*, COIN'09, pages 53–69, Berlin, Heidelberg, 2010. Springer-Verlag.

[29] Joey Sik-Chun Lam, Frank Guerin, Wamberto Vasconcelos, and Timothy J. Norman. Engineering societies in the agents world ix. chapter Coping with Exceptions in Agent-Based Workflow Enactments, pages 154–170. Springer-Verlag, Berlin, Heidelberg, 2009.

[30] Donghui Lin, Huanye Sheng, and Toru Ishida. Interorganizational workflow execution based on process agents and eca rules. *IEICE - Trans. Inf. Syst.*, E90-D(9):1335–1342, September 2007.

[31] Bertram Ludäscher, Mathias Weske, Timothy McPhillips, and Shawn Bowers. Scientific workflows: Business as usual? In Umeshwar Dayal, Johann Eder, Jana Koehler, and Hajo Reijers, editors, *7th Intl. Conf. on Business Process Management (BPM)*, LNCS 5701, Ulm, Germany, 2009.

[32] Zongwei Luo, Amit Sheth, Krys Kochut, and Budak Arpinar. Exception handling for conflict resolution in cross-organizational workflows. *Distrib. Parallel Databases*, 13(3):271–306, May 2003.

[33] Zongwei Luo, Amit Sheth, Krys Kochut, and John Miller. Exception handling in workflow systems. *Applied Intelligence*, 13(2):125–147, August 2000.

[34] Anna Malinova and Snezhana Gocheva-Iliev. Using the business process execution language for managing scientific processes. *Information Technologies and Knowledge*, 2:257–261, 2008.

[35] Paolo Missier, Satya Sanket Sahoo, Jun Zhao, Carole A. Goble, and Amit P. Sheth. *Janus*: From workflows to semantic provenance and linked open data. In *IPAW*, pages 129–141, 2010.

[36] Pierre Mouallem, Daniel Crawl, Ilkay Altintas, Mladen Vouk, and Ustun Yildiz. A fault-tolerance architecture for kepler-based distributed scientific workflows. In *Proceedings of the 22nd international conference on Scientific and statistical database management*, SSDBM'10, pages 452–460, Berlin, Heidelberg, 2010. Springer-Verlag.

[37] Hernâni Mourão and Pedro Antunes. Supporting effective unexpected exceptions handling in workflow management systems. In *Proceedings of the 2007 ACM symposium on Applied computing*, SAC '07, pages 1242–1249, New York, NY, USA, 2007. ACM.

[38] Hernâni Mourão and Pedro Antunes. Workflow recovery framework for exception handling: Involving the user. In *CRIWG*, pages 159–167, 2003.

[39] Adrian Paschke. Rule responder hcls escience infrastructure. In *Proceedings of the 3rd International Conference on the Pragmatic Web: Innovating the Interactive Society*, ICPW '08, pages 59–67, New York, NY, USA, 2008. ACM.

[40] Wei Ren, Gang Chen, Zhonghua Yang, Junhong Zhou, Jing-Bing Zhang, Chor Ping Low, David Chen, and Chengzheng Sun. Semantic enhanced rule driven workflow execution in collaborative virtual enterprise. In *10th International Conference on Control, Automation, Robotics and Vision, ICARCV 2008, Hanoi, Vietnam, 17-20 December 2008, Proceedings*, pages 910–915. IEEE, 2008.

[41] Satya S. Sahoo, Amit Sheth, and Cory Henson. Semantic provenance for escience: Managing the deluge of scientific data. *IEEE Internet Computing*, 12(4):46–54, July 2008.

[42] Yogesh Simmhan and Roger Barga. Analysis of approaches for supporting the open provenance model: A case study of the trident workflow workbench. *Future Gener. Comput. Syst.*, 27(6):790–796, June 2011.

[43] Yogesh L. Simmhan, Beth Plale, and Dennis Gannon. Query capabilities of the karma provenance framework. *Concurr. Comput. : Pract. Exper.*, 20(5):441–451, April 2008.

[44] Mirko Sonntag, Dimka Karastoyanova, and Frank Leymann. The Missing Features of Workflow Systems for Scientific Computations. In *Proceedings of the 3rd Grid Workflow Workshop (GWW), Software Engineering Conference, GI-Edition Lecture Notes in Informatics (LNI), P-160*, pages 209–216. Gesellschaft für Informatik e.V. (GI), February 2010.

[45] Rafael Tolosana-Calasanz, José A. Bañares, Omer F. Rana, Pedro Álvarez, Joaquín Ezpeleta, and Andreas Hoheisel. Adaptive exception handling for scientific workflows. *Concurr. Comput. : Pract. Exper.*, 22(5):617–642, April 2010.

[46] Yi Wang, Minglu Li, Jian Cao, and Feilong Tang. Flexible services composition based on eca rule in grid. In *Communications and Networking in China, 2007. CHINACOM '07. Second International Conference on*, pages 181 –185, aug. 2007.

[47] Hans Weigand, Willem-jan Van Den Heuvel, and Marcel Hiel. Rule-based service composition and service-oriented business rule management. *Business*, page 1âĂŞ12, 2008.

[48] Ustun Yildiz, Adnene Guabtni, and Anne H. H. Ngu. Business versus scientific workflows: A comparative study. In *Proceedings of the 2009 Congress on Services - I*, SERVICES '09, pages 340–343, Washington, DC, USA, 2009. IEEE Computer Society.

[49] Mark H. Ellisman Thomas Fahringer Geoffrey Fox Dennis Gannon Carole A. Goble Miron Livny Luc Moreau Jim Myers Yolanda Gil, Ewa Deelman. Examining the challenges of scientific workflows. *IEEE Computer*, 40(12):24–32, 2007.