Ontology for processing service orchestration

Mingda Zhang
State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing
Wuhan University
Wuhan, China

Joshua Lieberman
Center for Geographic Analysis
Harvard University
Cambridge, USA

Peng Yue*
School of Remote Sensing and Information Engineering
Wuhan University
Wuhan, China
pyue@whu.edu.cn

***Abstract*—With the advancement of Web service technologies, more and more spatial data and analysis functions are available on the web. Scientific workflows are widely used to orchestrate services to solve complex geospatial problems. The ontology for processing service orchestration (OPSO) is developed to empower domain experts to explore large datasets and complex processing workflows iteratively and collaboratively. OPSO represents three aspects of workflows: template, instance and provenance. The geoprocessing workflow tool named GeoJModelBuilder is extended to support OPSO, and an example will be used to demonstrate how the proposed ontology works.**

Keywords—geospatial ontology, scientific workflow, Web processing service

I. INTRODUCTION

Web service technologies have significantly promoted the wide sharing of distributed geospatial data and analysis functions[1, 2]. The forward workflows are extensively used to orchestrate services to solve complex geospatial problems [3, 4]. However, it is still difficult to find appropriate services to construct a workflow, and reuse or refine existing workflows to solve similar problems. It is important to describe workflow in a way that makes them more adaptable, sharable, reusable and traceable.

The Common Workflow Language (CWL) [5] is used to describe command line tools and connect them to construct workflows, which focuses on composing commands. The Web Services Business Process Execution Language (WS-BPEL) [6], commonly known as BPEL, is a widely used standard for chaining Web services. It is a process-centric language. Open Provenance Model for Workflows (OPMW) [7] is an extension of W3C PROV Ontology [8], Open Provenance Model (OPM) [9] and P-plan [10] to represent workflow provenance and template in general domain. The existing works either describe parts of the workflow aspects or are not service oriented.

The Semantic Web is an extension of the World Wide Web that enables well-defined knowledge understood and processed by computers [11]. It provides an effective approach to share and connect information. Semantic Web consists of many technical standards, including Web Ontology Language (OWL), Resource Description Framework (RDF), SPARQL (SPARQL Protocol and RDF Query Language) and so on. OWL is used to define basic concepts and relations among them. RDF is a standard model for data representation in form of "subject-predicate-object" triples. SPARQL is an RDF query language. This paper introduces the ontology for processing service orchestration (OPSO), which is developed to empower domain experts to explore large datasets and complex processing workflows iteratively and collaboratively using Sematic Web technologies.

OPSO represents three aspects of workflows: template, instance and provenance. Workflow template allows users to describe what kind of data and processing are needed in a workflow in an abstract way. Workflow instances are built when workflow templates are matched to distributed services, which are executable. Workflow provenance is generated during the workflow execution. It records data used, process steps and data dependencies between processes, which could not only document how the results come but also help refine or modify the workflow template and instance. There are also some other features in OPSO, such as branching control capability, spatial-temporal constraints and service orientation.

A geoprocessing workflow tool named GeoJModelBuilder [12] is extended to support OPSO. It allows users to generate workflow template and instance in a drag-and-drop way, and capture provenance information during workflow execution. The workflow template, instance and provenance could be exported to and reloaded from RDF files encoding in OPSO.

The remainder of the paper is organized as follows. Section II introduces workflow conceptual model. Section III presents the ontology for processing service orchestration. The extension to GeoJModelBuilder and a use case are described in Section IV. Section V provides the conclusions and pointers to the future work.

II. WORKFLOW CONCEPTUAL MODEL

A process is comprised of several input ports, output ports and a functional behavior, which could be abstracted as Input-Process-Output (IPO) form. A workflow consists of a couple of processes, and relations among them including data flows and control flows. OPSO follows the data-centric approach. Each process in workflow consumes data from input ports and produces data on output ports. Data flows, i.e., data dependencies, are built by linking between the output ports from source processes and inputs ports from the target processes. The control flow is treated as a special kind of data flow, which is associated with a controller.

This work is funded by China Scholarship Council.

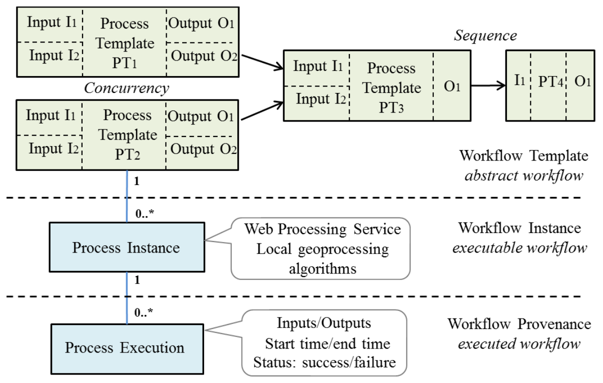

Fig. 1. Workflow conceptual model

There are three kinds of processes corresponding to different workflows (shown as Fig.1). They have different levels of abstraction. Process template describes the functionality of a process and how the outputs are derived from the inputs on a generic level. While, process instances refer to the concrete geoprocessing resources, such as processes in Web Processing Service (WPS) and local geoprocessing functions. Process instance has complete definition of inputs and outputs, which may include specific data type and formats. One process template may have several process instances. Workflow instance is generated from workflow template by mapping the process templates to concrete resources, which is ready to execute. The business logics and resource usage are separated by these two levels, which has advantages of logical consistency, physical separation and dynamic adaptation. Semantic constraints could be added to the workflow template to enable the automatic or semi-automatic generation of workflow instance. Process execution information, i.e., provenance information, records the input data and output data, execution start time and end time, and the execution status (success or failure). An instance could be executed many times. So a process instance has several process executions.

The control flow logics include sequence, concurrency, branching and loop. The data flows already have the capabilities to express the sequence and concurrency logics (shown in Fig. 1). *Controllers* are introduced into workflows to represent branching. The controller is a special process. Its function is to determine the condition is true or false. It is like a switch assigned to a data flow. For example, in Fig. 2, the *Condition* determines which one, P_2 or P_3, will be executed after the P_1. If the condition is false, the data flow identified by $P_1O_1\#P_3I_2$ will be valid, which means P_3 will be executed using the output of P_1 as its second input. Otherwise, it is P_2 that follows P_1. Then, the data flows back to P_1 again, which forms a loop. In such situation, the controller will be invoked repeatedly until the condition becomes false. The object of the condition may be assigned by users or come from the result of previous process node. In spatial domain, the condition could be more complex. The execution order of process in a workflow is determined by data flows as well as controllers if exist.

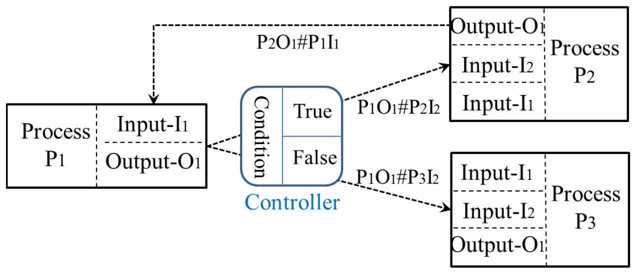

Fig. 2. Control flow represented by data flow and controller

III. ONTOLOGY FOR PROCESSING SERVICE ORCHESTRATION

OPSO is based on OPMW and has some new features: three level representations of workflows, branching capabilities, spatial-temporal constraints and service-orientation. Fig.3 shows a high level view of OPSO. The existing vocabularies used in OPSO are listed in Table.1, which include SPIN Modeling Vocabulary [13], and SDWGEO [14]. SPIN provides the ability to encapsulate SPARQL queries into RDF, which is used to enable branching capabilities in OPSO. SDWGEO aims at spatial representation on the Web.

Table 1. Onotology vocabularies used in OPSO

Prefix	Namespace	Description
OPMW	http://www.opmw.org/model/OPMW/	Starting point, has abilities to represent workflow template and execution
SDWGEO	http://geosemweb.org/sdwgeo	Spatial representation
SPIN	http://spinrdf.org/spin#	Embeds SPARQL into RDF, represents conditions

A. Three level representation

In Fig.3, the light green figures represent classes from OPMW. The classes, *WorkflowTemplateArtifact*, *WorkflowTemplateProcess* and *WorkflowTemplate*, could be used to represent workflow template. *WorkflowExecutionArtifact*, *WorkflowExecutionProcess* and *ExecutionAccount* are used to represent workflow provenance. *ExecutionAccount* is a sub-class of *prov:Bundle* which is a set of provenance descriptions. The Three object properties, i.e., *correspondsToTemplate*, *correspondsToTemplateArtifact*, and *correspondsToTemplateProcess*, are used to associate the execution information with the templates.

The instance-related classes in OPSO inherit from template-related classes in OPMW, because the instance could be treated as specific ones of template, and many properties could be reused. Web processing service is a specific resource of process instance. The *opso:coresspondsToInstance** series object properties are added to represent the relations between the executions and instances, *opso:hasInstance** series for relations between templates and instances.

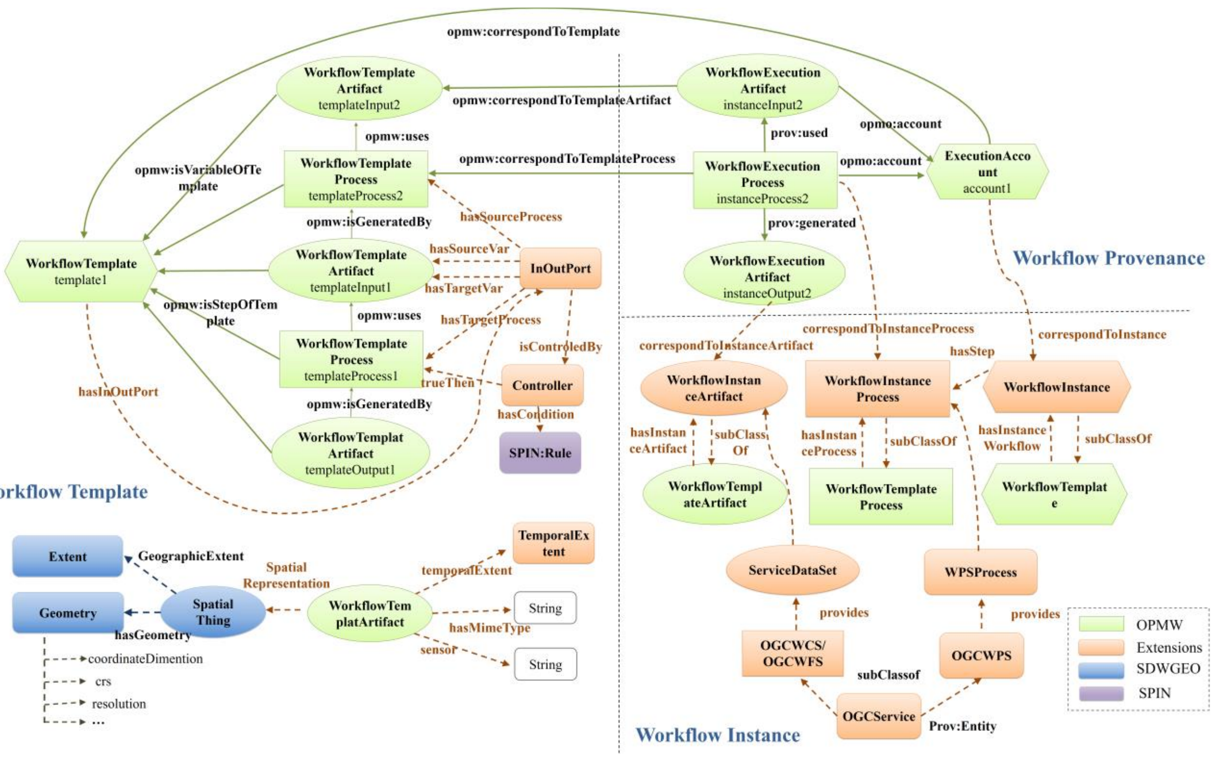

Fig. 3. Ontology for processing service orchestration (OPSO)

B. *Data flow & control flow*

A new class named *opso:InOutPort* is added to represent the data flow explicitly, which has four associated properties: *opso:hasSourceProcess*, *opso:hasSourceVar*, *opso:hasTargetProcess*, *opso:hasTargetVar*, to indicate how one process depends on another.

As to control flow, one more class named *opso:Controller* is used, which is related to SPIN rules by an object property named *opso:hasCondition*. The SPIN Modeling Vocabulary is proposed to specify rules and logical constraints using SPARQL, which provides the ability to encapsulate SPARQL queries into RDF.

C. *Spatial-temporal constraints*

Workflow template is the "recipe" required to produce a data artifact. So what kinds of resources are needed should be described in detail. For spatial data, the spatial and temporal characteristics are distinctive features, which help constrain the spatial-temporal extent of the data. SDWGEO is used as spatial representation. An object property named *opso:spatialRepresentation* is provided to combine *opmw:WorkflowTemplateArtifact* and *sdwgeo:SpatialThing*. A class named *opso:TemporalExtent* is added to restrict the temporal extent of expected data. Besides, spatial data has various formats. *MimeType*, *geometricObjects* (for vector data: point, curve, surface, etc.) and *cellGeometry* (for coverage: point, area, voxel, etc.) will also be used as spatial representation. As for coverage data, the platform, sensor, dimension and resolution are also useful information. Most aforementioned aspects are referred to ISO 19115.

D. *Service orientation*

A WPS server (*opso:OGCWPS*) usually provides many processes (*opso:WPSProcess*). Every process in WPS plays a role of process instance. It is the same with data services. *opso:OGCWCS* and *opso:OGCWFS* could provide *opso:ServiceDataset*.

IV. A USE CASE AND PROTOTYPE TOOL

The use case is about detecting changes in the water coverage area (shown in Fig.4). Band 1 and 2 of MODIS image go through the Normalized Difference Vegetation Index (NDVI) calculation, binarization, and rendering processes to derive the water coverage for each period. The two images of different periods are mixed using blending process to generate a thematic flood image that reflects changes in the water coverage area.

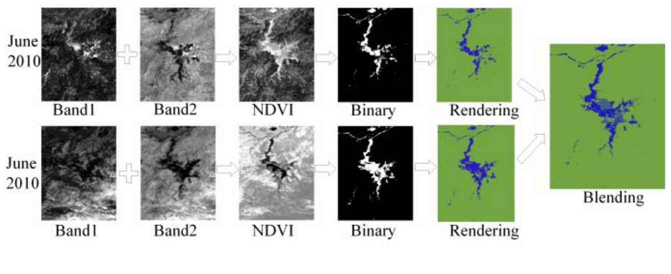

Fig. 4. Scientific workflow for water change area detection

GeoJModelBuilder is an open source geoprocessing workflow tool. It is able to integrate Sensor Web, geoprocessing services, and OpenMI-compliant model components for environmental monitoring and integrated modelling. It is extended to support semantic functionalities. The source code for new version is moved to GitHub (https://github.com/geoprocessing/GeoJModelBuilder). Fig.5 shows the transformation among workflow aspects and related functions provided by GeoJModelBuilder.

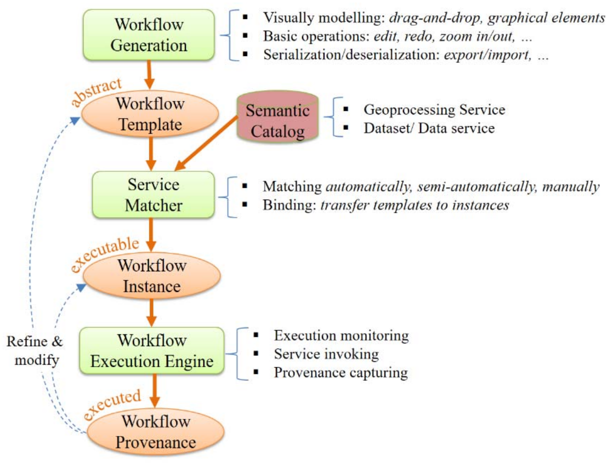
Workflow Generation
Visually modelling: *drag-and-drop, graphical elements*
Basic operations: *edit, redo, zoom in/out, ...*
Serialization/deserialization: *export/import, ...*
abstract
Workflow Template
Semantic Catalog
Geoprocessing Service
Dataset/ Data service
Service Matcher
Matching *automatically, semi-automatically, manually*
Binding: *transfer templates to instances*
executable
Workflow Instance
Refine & modify
Workflow Execution Engine
Execution monitoring
Service invoking
Provenance capturing
executed
Workflow Provenance

Fig. 5. Transformation among workflow aspects and related functions in GeoJModelBuilder

The GUI (Graphical User Interface) in GeoJModelBuilder is re-implemented based on Eclipse Graphical Editing Framework (GEF). GEF is a powerful MVC (Model-View-Controller) framework, which provides a standard way to create rich graphical Java client applications. Different graphical figures are used as building-blocks to generate a workflow visually. For example, rectangle represents the process, and eclipse represents input or output data. Fig.6 shows the constructed workflow in GeoJModelBuilder for the use case. The panel in the left lists the processes retrieved from a WPS server named GeoPW [2]. Any geoprocessing service that follows WPS 1.0 standard could be added into this panel easily. In the middle, it provides basic elements to construct a workflow, such as data, process, condition, data flow and control flow (*TrueThen* and *FalseThen*). GeoJModelBuilder allows users to drag these basic elements as well as WPS processes to build workflows in the right part. When using WPS process or Process element, a process template will be both generated. The difference is that the former will automatically bind an instance, i.e., WPS process, to the process template. Users may also be allowed to bind specific services to template manually.

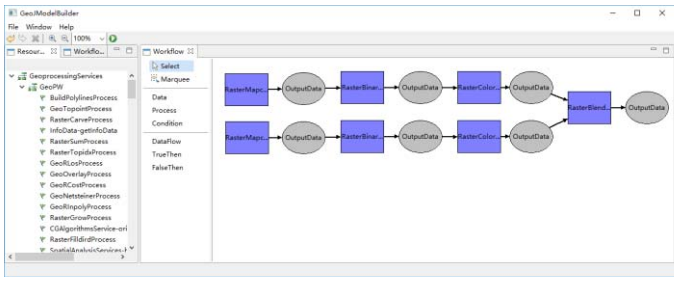

Fig. 6. Scientific workflow in GeoJModelBuilder

Once the elements in a workflow are bound to particular geoprocessing services and dataset, the workflow becomes executable. A workflow execution engine is integrated into GeoJModelBuilder. Its functionality is to invoke services orderly according to the data flows and control flows, monitor the execution and capture the provenance. For reusability, user can import and export the workflows from and to RDF files. Jena is leveraged to support the capabilities.

V. Conclusion and Future Work

This paper proposes a workflow language named Ontology for Processing Service Orchestration (OPSO). It represents workflows in three aspects: template, instance and provenance using Semantic Web technologies. OPSO enables users to build and share semantic workflow templates that describe data, processing, and control flow components, then match those "recipes" to appropriate distributed service components in order to build and execute workflow instance. Provenance information will be generated during the workflow execution, which could be used to easily refine templates or modify instances, e.g. for different inputs, regions, spatiotemporal aggregations, or product types.

In the future work, we will extend the HHypermap [15], a services registry and spatio-temporal search platform, to provide semantic descriptions of services, then come up with some approaches to match the workflow recipes to those concrete resources automatically and semi-automatically instead of manually.

Acknowledgment

The work was supported by Major State Research Development Program of China (2017YFB0503704), Hubei Science and Technology Support Program in China (2014BAA087), the Nature Science Foundation Innovation Group Project of Hubei Province, China (No. 2016CFA003), and China Scholarship Council.

References

[1] P. Yue, C. Zhang, M. Zhang, Z. Xi and L. Jiang. "An SDI Approach for Big Data Analytics: The Case on Sensor Web Event Detection and Geoprocessing Workflow". IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 2015, 8(10), 4720-4728.

[2] P. Yue, J. Gong, L. Di, J. Yuan, L. Sun, Z. Sun and Q. Wang, "GeoPW: Laying blocks for the geospatial processing web," Transactions in GIS, 2010, 14(6), 755-772.

[3] P. Yue, R. Ramachandran, P. Baumann, S. J. S. Khalsa, M. Deng, and L. Jiang, "Recent Activities in Earth Data Science [Technical Committees]," IEEE Geoscience and Remote Sensing Magazine, 2016. 4(4), 84-89.

[4] P. Yue, X. Guo, M. Zhang, L. Jiang and X. Zhai. "Linked Data and SDI: The case on Web geoprocessing workflows," ISPRS Journal of Photogrammetry and Remote Sensing, 2016, 114, 245-257.

[5] Common Workflow Language. Available from: http://www.commonwl.org/.

[6] Web Services Business Process Execution Language Version 2.0, 2007, OASIS.

[7] D. Garijo and Y. Gil, The OPMW-PROV Ontology, 2014, Available from: http://www.opmw.org/model/OPMW/.

[8] T. Lebo, S. Sahoo and D. McGuinness. PROV-O: The PROV Ontology, 2013, Available from: https://www.w3.org/TR/prov-o/

[9] L. Moreau, J. Freire, J. Futrelle, R. E. McGrath, J. Myers and P. Paulson, “The open provenance model: An overview”. in International Provenance and Annotation Workshop. 2008. Springer.

[10] D. Garijo and Y. Gil, The P-PLAN Ontology, 2014, Available from: http://vocab.linkeddata.es/p-plan/.

[11] T Berners-Lee, J. Hendler, and O. Lassila, “The semantic web,” Scientific american, 2001, 284(5), 28-37.

[12] P. Yue, M. Zhang and Z. Tan, “A geoprocessing workflow system for environmental monitoring and integrated modelling,” Environmental Modelling & Software, 2015, 69, 128-140.

[13] H. Knublauch, SPIN - Modeling Vocabulary, 2011, avaiable from https://www.w3.org/Submission/spin-modeling/

[14] SDWGEO, available from: http://geosemweb.org/sdwgeo.

[15] P. Corti, B. Lewis, T. Kralidis and J. Mwenda, Implementing an open source spatio-temporal search platform for Spatial Data Infrastructures, PeerJ Preprints, 2016,