Chapter 98
Formalizing Cross-Parameter Conditions for Geoprocessing Service Chain Validation

Daniel Fitzner
Fraunhofer IGD, Germany

ABSTRACT

Geoprocessing operations offered via web services provide the means for building complex web-based geospatial applications. Often, certain postconditions such as the spatial reference system, bounding box, schema or quality that hold on the output dataset after the execution of a geoprocessing service are determined and derived from the properties of the inputs passed to the service. Further, geoprocesses often hold preconditions that relate to more than one input, such as the requirement that all inputs must have the same schema. Within current process descriptions for geoprocessing operations, such conditions which we call cross-parameter conditions, can not be explicitly specified. In this paper, the author gives an approach to formalize such cross input-output and cross input parameter conditions in a rule-based language. Further, the author proposes an algorithm for deriving pre- and postconditions for a service composition or workflow out of the pre- and postconditions of the services involved, allowing a more automated handling of workflows in general.

INTRODUCTION

Geoprocessing tools are software components that input (geospatial) datasets; perform some sort of processing on the inputs and output (geospatial) datasets. Usually they perform some sort of calculation on the geometric attribute of the input data.

DOI: 10.4018/978-1-4666-2038-4.ch098

Such geoprocessing tools can either be generic and therefore reusable in a number of different application contexts or specific to an application case. Standard, desktop-based GIS such as ArcGIS (Ormsby, Napoleon, Burke, Groessl, & Bowden, 2004) usually come with a set of generic geoprocessing tools since – due to the generic nature of GIS – these tools should be reusable in different application contexts. For example, a standard GIS-geoprocessing tool that calculates a buffer around features can be reused in a number of different application contexts, e.g. for calculating a buffer around datasets representing cities, highways or anything else. To enable users to perform application specific calculations and to create application specific tools for reuse, standard GIS allow the combination of such generic geoprocessing tools. This means, the output of one tool is given as input to another (or the same) tool and so on and the whole chain then outputs datasets that can not be derived from any of the known data sources directly or from executing single generic geoprocessing tools.

For simple chains, involving only few generic geoprocessing tools, it is manageable to perform this process fully manual. This means, a human user executes a tool on some dataset, executes another tool on the resulting dataset and so on until the desired result is reached. However, standard desktop-based GIS, in order to enable reuse and automation of such chains of geoprocessing tools, further offer the possibility to store them such that they can later be retrieved and executed. In GIS terminology, such chains are often called *models* or *workflows*. Most GIS come with a user interface that allows users to graphically compose workflows of geoprocessing tools without the need for programming. Although the workflows in desktop GIS can typically be exported into some sort of script-file, the users composing them usually do not get in contact with program code. Additionally, GIS often offer the possibility to parameterize workflows, i.e., some of the input datasets or values (or all of them) are not stored with the workflow (e.g., as pointers in the script-file) but can be specified by users when executing the workflow. This has several advantages: First, storing the input data to a workflow would typically mean storing a pointer to a data source somewhere on hard drive (e.g., a file path) or in the current GIS-workspace. As soon as the location of the data or the workspace changes, the workflow becomes invalid. Second, the parameterization of the workflow allows building a simple user interface within the GIS that allows specifying the inputs without the need for editing or changing the workflow as it is stored. This means, when executing the workflow, users, who are probably not familiar with all the details of the workflow only see the entry points (the input signature) to the workflow without being confronted with the whole, probably quite complex processing encapsulated within.

With the advent of web service technology, the geoprocessing tools do not necessarily need to be integrated with the desktop GIS, but might be encapsulated within a web service. The W3C gives the following characterization of web services: “Web services provide a standard means for interoperating between different software applications, running on a variety of platforms and/or frameworks”[1]. The OGC Web Processing Service standard (Open Geospatial Consortium Inc., 2010) is the most widely used web service interface definition for geoprocessing. It defines a generic set of operations that need to be offered by any web service aiming to be compatible to the standard. These include operations for requesting service- and process metadata as well as operations for executing algorithms offered by the service. The overall goal of the WPS specification is to enable the reuse of server and client code. Libraries exist that allow service providers to expose quite easily their algorithms via the WPS interface (see e.g., the software provided by 52n[2]). Further, functionalities exist that allows users to invoke WPS, without the need for doing implementation work (see e.g., Foerster & Schäffer, 2007). Several works exist that discuss how to wrap geoprocessing operations implemented within GIS with a web service interface. For example (Brauner, 2008) gives a methodology on how to wrap geoprocessing tools implemented in GRASS-GIS (Neteler & Mitasova, 2002) behind a WPS interface or (Díaz, Costa, Granell, & Gould, 2007) show how to model geoprocessing services for water resource management applications.

Related Content