Distributed Services Technology for Earth Science Data Processing

Ken Keiser, Rahul Ramachandran, John Rushing, Helen Conover, and Sara Graves

Information Technology and Systems Center
University of Alabama in Huntsville
Huntsville, AL - 35899

The Information Technology and Systems Center (ITSC) at the University of Alabama in Huntsville (UAH) is investigating the use of distributed services technologies for complex data processing solutions. With the advent of web services it seemed timely to investigate how some of these technologies could be utilized for data processing problems that arise when working with Earth Science data sets. Some common characteristics of these data sets are that the data is typically structured as image data, is spatially referenced, and data files are large. In this paper we introduce the ITSC concept of **Simple Services**, and provide results of a prototype chaining these distributed services to handle the processing of large data sets.

The current trend in web services is the use of SOAP (simple object access protocol), WSDL (web service description language), UDDI (universal description, discovery, and integration) and associated technologies to provide language and system independent distributed services [9]. This infrastructure uses underlying extensible markup language (XML) technology as the packages for transferring method requests and corresponding parameters and data.

The SOAP-based technologies appear to offer great possibilities of allowing the integration of tools and services across heterogeneous platforms. While promising, this interoperability still comes at some expense of implementation. In this case, each service host must provide SOAP-enabled processing to interpret the SOAP envelopes and then handle directing requests to the correct service. Additionally, there are some compliance constraints that need to be supported by the services themselves in terms of their programming interfaces and how they are registered/advertised. As an over-simplification, this technology appears to be well suited for transactional, text-based exchanges. It does not, however seem to adequately address handling large (binary) data transfers. Some approaches have been identified to allow the encoding of binary objects [8] within XML but this approach does not appear to be optimized for working with large non-text data sets. Alternate approaches to distributed services have been researched by ITSC to resolve the large data set problems.

ITSC began investigating the use of distributed services for use with data mining, subsetting, image processing, image/map generation and other spatially oriented data applications [3,7]. Image processing operations, for instance, seem to lend themselves well to being broken down into discrete processes that can be combined to provide a more complex solution [6]. This observation then directed the ITSC focus towards investigations of service chaining. For the purpose of this discussion, we identified two main architectures that might support the chaining of services: (1) using a central controlling mechanism that handles the serial execution of multiple services in an order defined to be the chain (see Figure 1-a) and (2) an approach of scripting/scheduling a sequence of services that will be executed in a specific order, but the sequential execution is controlled by the services themselves rather than an intermediary application (see Figure 1-b). Additionally, the ITSC team identified a set of assumptions considered to be necessary to allow for a good flow of data through processes and to allow for the simplest architecture of services: (1) extra file I/O (reading and writing of data files) should be avoided – implying that data should be streamed, when possible, between services;

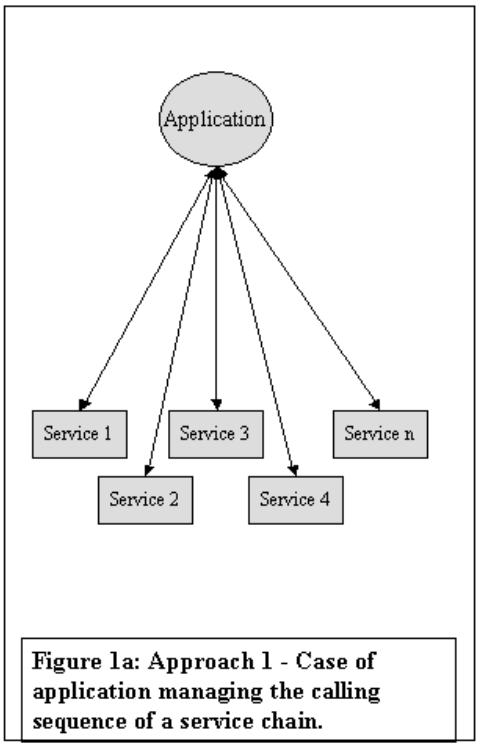

Figure 1a: Approach 1 - Case of application managing the calling sequence of a service chain.

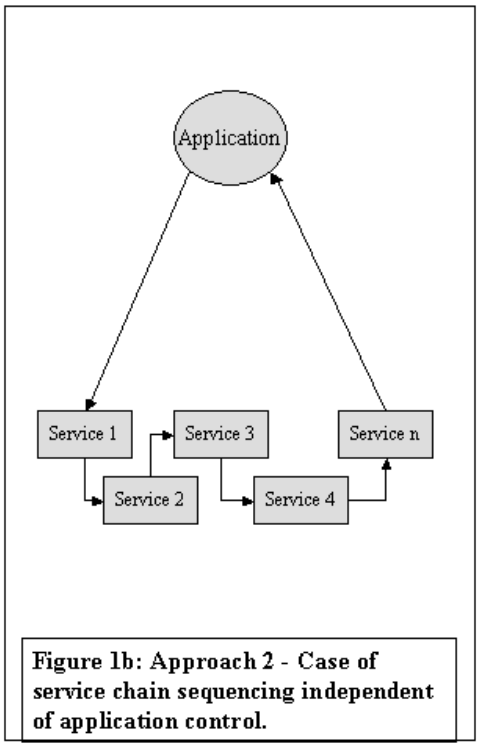

Figure 1b: Approach 2 - Case of service chain sequencing independent of application control.

(2) the services should be autonomous, implying that they can stand alone and have no application-specific operational constraints or requirements; (3) there should be no constraints or dependencies in terms of programming languages or hardware platforms.

To examine the chaining approaches within these assumptions more closely, ITSC focused initially on image processing services to evaluate how best they could be combined to affect an efficient flow of data between them. Chaining approach (1) (Figure 1a) creates obstacles to an efficient data flow through the chain in that the data would be transferred back-and-forth between each service and the controlling application, thereby creating what appears to be a lot of unnecessary input/output (I/O) processing. Having identified this potential problem, ITSC researchers focused on investigating chaining approach (2) (Figure 1b) that flows data from service-to-service without application intervention.

This investigation led ITSC to define a ***Simple Service*** **architecture** (Figure 2). Considerations for a Simple Service include:

- A Simple Service is autonomous in that it is functional by itself and knows nothing of the environment that is using it.
- By using the URL protocol to access Simple Services and stream back results, they can be logically treated as opening a file, even while passing in processing parameters in the URL specification.

Chaining Simple Services

Because a Simple *S*ervice can be viewed logically as a file, multiple Simple Services can be chained together by letting the file to be opened by one service be the URL of the next service (with the argument of the next file or service). We view this as the *service-is-a-file* paradigm for service chaining. Figure 3 illustrates this approach.

There is a problem with chaining multiple services in this way, however. Chaining is activated in this model by nesting the sequence of URLs with each subsequent service indicated by a "file=" argument. The URL syntax of this nesting unfortunately

gets ugly after more than one level and in fact does not appear to be syntactically possible with the HTTP Get/Put protocol.

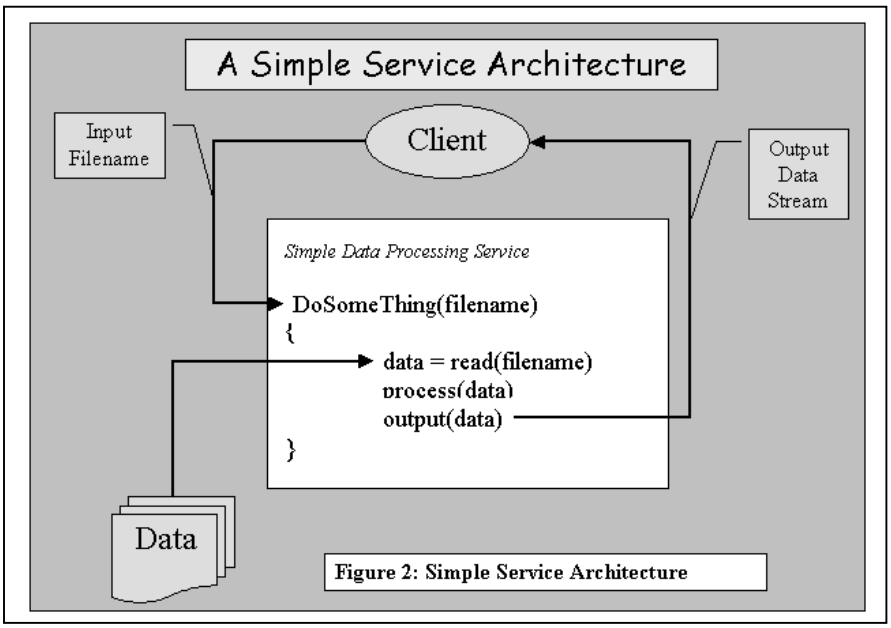

Figure 2: Simple Service Architecture

To get around this problem a proxy generator (or registry) can be used by the application to register the sequence of services to be chained. The proxy generator is similar to but simpler than a "mediating services" approach that serves to aggregate and manage services for chaining [1]. With this approach, the "file=" argument passed to each service is in reality the address of the proxy generator which merely redirects the stream to the next registered service with the appropriate arguments. This approach still avoids any additional responsibility on the part of the individual services, but rather puts the burden on the application to define and register the chaining path. Figure 4 illustrates an example chain using the proxy generator approach.

The proxy generator approach is being used in an ITSC prototype application using *service-is-a-file* chained services. The prototype's main application is implemented using Java Server Pages for the user interface components. The Proxy Generator is implemented as a Java Servlet. The individual image processing services are implemented as a distributed mix of Java servlets, Perl scripts and C programs with Perl wrappers to handle the HTTP communications, on both Windows and Linux systems. The initial prototype user interface is minimal in the sense that it provides the ability to specify up to three services and the original image file with no way to specify service parameters. The interface additionally displays the original and final images (Figure 5), demonstrating the result of the chained services.

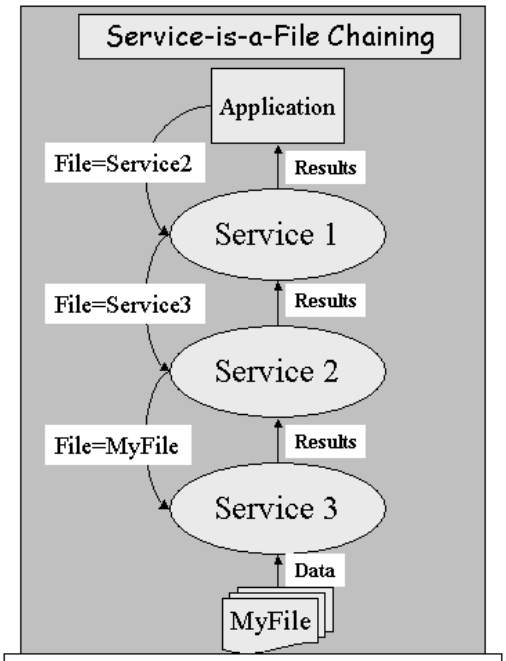

Figure 3: *Service-is-a-File* Concept of Chaining Simple Services

The Prototyped Simple Services.

All of the services implemented for the prototype followed the Simple Service architecture in that they take, at a minimum, the name of a target file as input. This file is opened via URL access. The contents of the file are processed and then output on the HTTP response stream of the service.

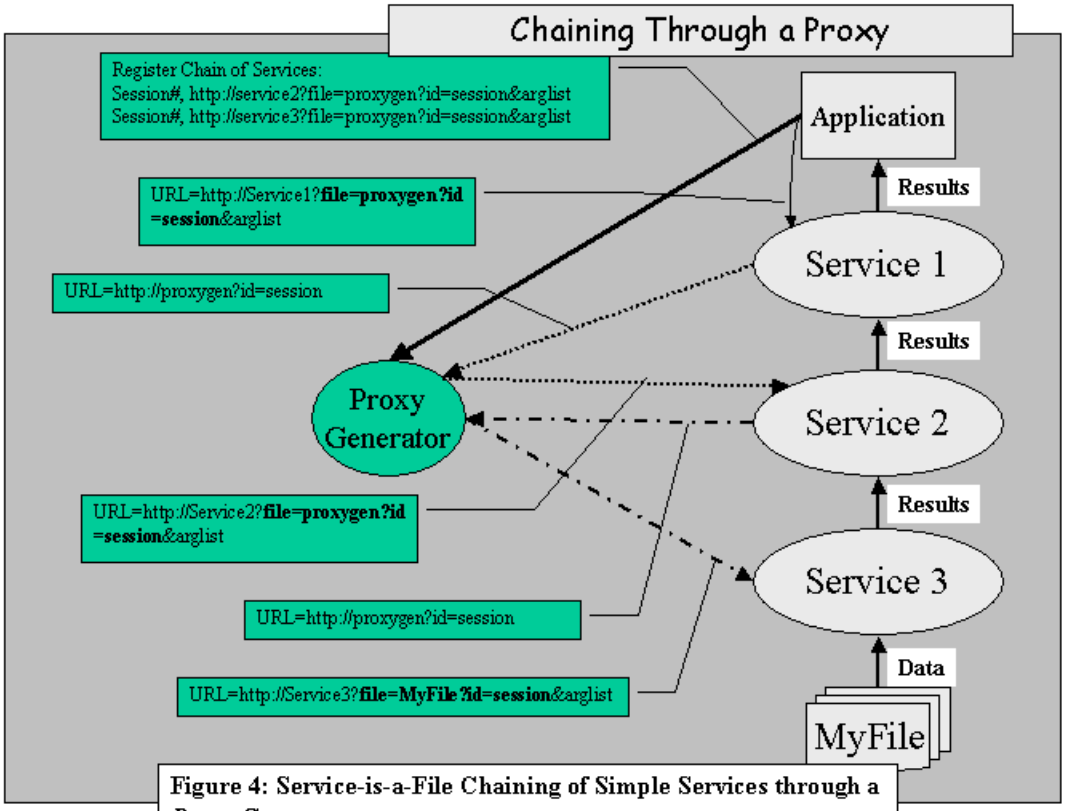

Figure 4: Service-is-a-File Chaining of Simple Services through a *Proxy Generator*

An initial list of services consists of the following:

- Crop – crops the input image to specified dimensions in image units.
- Rectangle – draws a rectangle of a specified color in the image (called "black out" on the demo interface since the color was hard-coded to black).
- Format – reformats the input image to a specified output type.
- Resample – resamples/resizes the input image to specified dimensions.
- Geo_crop – crops the input image based on geographic coordinates.

Limitations and Plans

While this prototype has been successful in keeping the services "simple" as defined above and at the same time providing some useful functionality, this technique will need some enhancements for use with more sophisticated applications trying to solve more complex problems such as data mining. Not all data comes in the form of the image files used for this initial implementation. A way to pass heterogeneous data between the services is necessary, while at the same time keeping the services simple and autonomous.

ITSC data processing research has been primarily focused on heterogeneous Earth Science data sets and most recently on ways of interoperating with these data sets. With support from the NASA Earth Science Technology Office, our lab has developed the Earth Science Markup Language (ESML) [http://esml.itsc.uah.edu] to provide true access level interoperability across geospatial data sets [5]. An ESML description is an external metadata file containing structural and semantic information to enable an application to intelligently read associated data file(s).

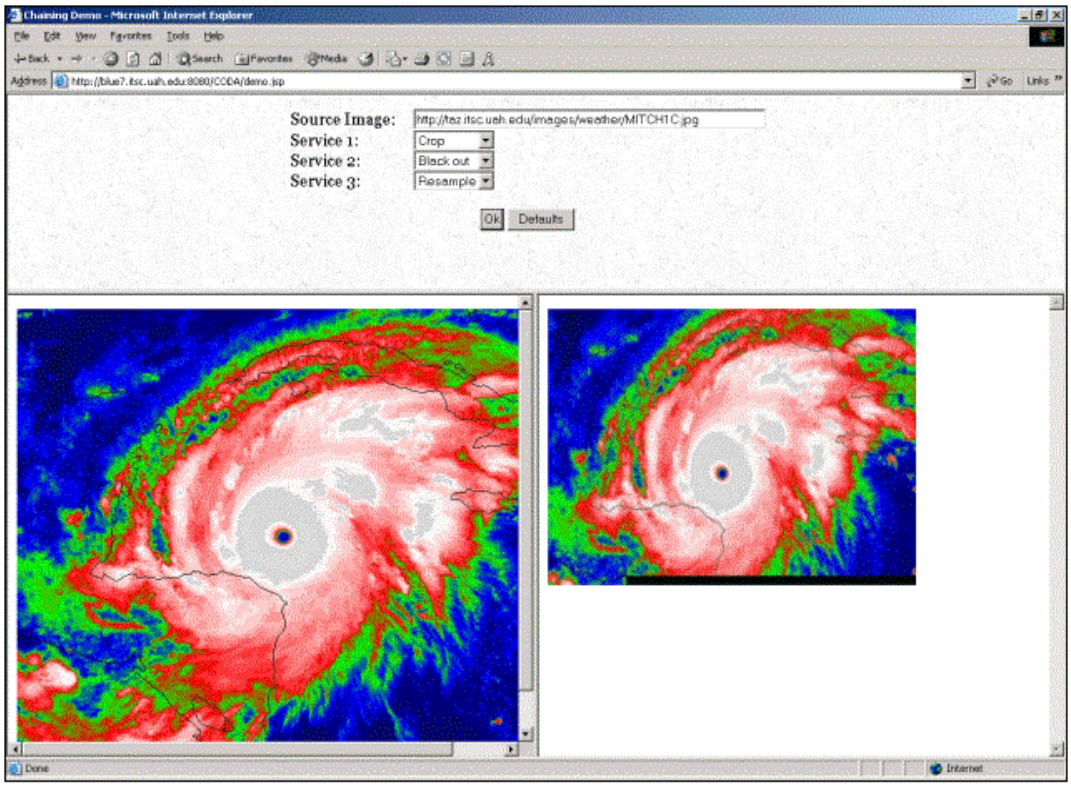

Figure 5: Image Processing Prototype Application using Service-is-a-File Chaining of Simple Services

In future research ITSC plans to investigate the incorporation of ESML into the Simple Service architecture. This will provide a way for services to discover the output format of the services they call as well as ways for the driving application to specify dynamic data translations as the data flows between services.

Some official definitions of a web service suggest that all services should be self-describing. Typically this includes detailing the calling parameters but also the possible outputs. The OpenGIS Consortium uses the approach of all services providing a "capabilities" document that describes what service is provided. The ***service-is-a-file*** paradigm that we are using would be well served by using an ESML description of output data formats as (at least) part of the "capabilities" document for chain-able services. This approach on the surface seems to contaminate the "simple" aspect of the architecture in that a service capable of reading an ESML description would certainly have additional dependencies on outside libraries and documents. An alternative would be to have ESML-enabled data adaptors/filters that can be chained in between other services to act as data-mediators between autonomous services. ITSC is continuing to study these options.

Conclusions

We have demonstrated that within the constrained Simple Service architecture the ***service-is-a-file*** paradigm of chaining appears to work well and handles data in an efficient manner. The Simple Service architecture achieves the original goal of having services that are autonomous (can be used outside of any particular architecture) and have no additional dependencies such as external libraries or formatting knowledge. It is doubtful that the *service-is-a-file* paradigm will scale well for larger, more complex applications but perhaps the model can evolve for future service chaining investigations where services are needed to work with large distributed data volumes.

ITSC is also using this approach to implement a series of OpenGIS Consortium (OGC) Web Map Servers and related services [4]. OGC Map and Data services seem to be good candidates for implementing this chaining approach since there are several discrete operations that are applied to the data stream in different configurations depending on the client's request. Once implemented these discrete operations (services) can then be reused to support multiple instantiations of these types of request servers. Likewise the PM-ESIP Custom Order and Data Analysis (CODA) servers are comprised of these discrete services that when chained together fulfill the data order and visualization requests [2]. After additional testing and verification ITSC should be able to make these services available to the public as services that can be applied to large online data stores, such as the ITSC-administered Passive Microwave Open Network of Data (POND) and other distributed data sets. The ESML data interchange layer integrates nicely into this approach by providing interoperable data filters for the widely variable Earth Science data formats. Together these technologies offer good solutions for providing distributed data and services through web-based applications.

References

[1] Alameh N., "Scalable and Extensible Infrastructures for Distributing Interoperable Geographic Information Services on the Internet", PhD Dissertation Submitted to MIT Libraries, 2001

[2] Conover, Helen, Sara J. Graves, Ken Keiser, Chuck Pearson, Rahul Ramachandran, "A Next Generation Information System for Earth Science Data", The International Symposium on Optical Science, Engineering and Instrumentation, Denver, 1999.

[3] He, Yubin, Rahul Ramachandran, Udaysankar J. Nair, Ken Keiser, Helen Conover, Sara J. Graves, "Earth Science Data Mining and Knowledge Discovery Framework", SIAM International Conference on Data Mining, Arlington, VA, Apr. 11 - 13, 2002

[4] OGC 2002 OGC Web Service Initiative 1 (OWS1) Baseline Documents, WWW Document, http://ip.opengis.org/ows1/docIndex.html

[5] Ramachandran, Rahul, et al., "Earth Science Markup Language", 17th Conference on Interactive Information and Processing Systems for Meteorology, Oceanography, and Hydrology, 81st American Meteorological Society (AMS) Annual Meeting, Albuquerque, NM, January, 2001.

[6] Ramachandran, Rahul, Helen Conover, Sara J. Graves, Ken Keiser, Sunil Movva, Steve Tanner "Flexible Earth Science Data Mining Architecture", Fourth Workshop on Mining Scientific Datasets, Seventh ACM SigKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, August 26, 2001.

[7] Ramachandran, Rahul, Helen Conover, Sara J. Graves, Ken Keiser, "Algorithm Development and Mining (ADaM) System for Earth Science Applications", Second Conference on Artificial Intelligence, 80th AMS Annual Meeting, Long Beach, CA, Jan., 2000.

[8] Salz, Rich, "Transporting Binary Data in SOAP", XML.COM Endpoints Article, August, 2002: http://www.xml.com/pub/a/2002/08/28/endpoints.html

[9] W3C Draft SOAP Standard Specification: http://www.w3.org/TR/2001/WD-soap12-20010709/