

## **eServices Project Develops the Web Service Resource Framework (WSRF) in C and Python to Support Scientific Computing**

The eServices project, a collaboration between researchers at Argonne National Laboratory and Lawrence Berkeley National Laboratory, has released the first full-featured WSRF implementations in C and Python. This release is vital to support the adoption of Web Services-based Grids by the scientific community. The C and Python implementations are fully compatible with the Java WSRF implementation. The Globus Toolkit 4.0 release will include both Python and C tools to automatically generate code from a WSDL file. It will also support for WS-Addressing and WS-Security and will be compliant with the industry standard WS-Interop Basic Profile 1.0. By developing WSRF technology in languages widely used in the scientific computing world, focused on high performance and ease of integration, this project helps ensure the relevance of WSRF to the scientific computing community.

Besides language compatibility, the WSRF-C implementation offers many performance improvements over the Java implementation. For example, using a Java client requires the startup of the Java Virtual Machine (JVM), an expensive operation for a command-line client that is simply trying to start a job. Using the C client instead improves job submission rates by an order of magnitude. Factors responsible for this improvement include direct process execution instead of JVM startup, efficient XML pull-and-push parsers integrated with WSRF-C, and the ability to leverage XIO, an efficient transport layer developed as part of the Globus Toolkit that enables high-performance transfer by providing such features as security caching. In addition, a scientist can take advantage of the nonblocking stubs generated in C that provide a completely asynchronous API, allowing overlapping computation and communication calls. This feature is critical for high performance especially in nonthreaded applications such as many legacy codes. Moreover, these performance features are especially useful for scalability of volume transfers: in tests sending and receiving thousands of messages, we have seen order of magnitude improvements over the Java implementation.

The Python implementation offers similar benefits: a smaller memory footprint and significantly faster startup time compared to the Java implementation. The benefits derive mainly from the differences between the Java VM and the Python interpreter. The Python implementation is also significantly faster for doing message-level security because of the use of simple Python bindings to the OpenSSL toolkit written in C. In addition, the Python implementation includes a prototype version of a tool to automatically wrap a command-line application and provide a WSRF interface to it. This allows scientists to interact with a remote code in exactly the same way they would with the local version.

Encouraging user testimonies are beginning to come in as the infrastructure matures. An early version of the C implementation (developed in the context of Open Grid Services Infrastructure, a precursor to WSRF) is used by the NEESgrid project, a testbed for building the national virtual collaboratory for earthquake engineering to provide telecontrol capabilities. Since the NEESgrid programs are typically written in C,

MATLAB, or FORTRAN, it was vital to provide a C language client to experiment control programs. Furthermore, in this context, performance and process startup time are important. Infrastructure developed for this project has been in production operation since October 2004.

In the past few months, the OGS-C tools have been updated to reflect the WSRF specification and have gained in features, stability, and performance. Built on the C WS Core (which includes the WSRF-C implementation), the globusrun-ws client has become the job submission tool in the GT 4.0. It provides fast startup, good performance, interoperability with the GRAM services (written in Java), and all the required features that are part of the job submission process, including notification consumption and credential delegation. The availability of C client spawned other efforts. For example, we are experimenting with using C WS Core and C WSRF to embed a WSRF-enabled computational steering tool into GridFTP. This allows users to monitor transfers and tweak parameters on the fly to affect performance changes.

The current WSRF-C implementation is in the Globus Toolkit 3.9.5 release representing the beta code. The final release for the Globus Toolkit 4.0, which will include the WSRF-C implementation, is planned for April.