

A File System Service for the Venice Service Grid

Markus Hillenbrand, Joachim Götze, Paul Müller
University of Kaiserslautern
D-67653 Kaiserslautern, Germany
{hillenbr, j_goetze, pmueller}@informatik.uni-kl.de

1 Motivation

The Venice Service Grid [7] (in short just Venice) provides a framework for building applications on top of widely distributed services. As a Service Grid, Venice abstracts from underlying hard- and software, hides the complexity of its resources and focuses on the end-user by providing intuitive means for service access. In order to be lightweight, Venice is easy to deploy (i.e. quick service creation and deployment), easy to maintain (i.e. tool support for management, monitoring, and configuration at runtime) and easy to use (i.e. no installation for end-users, graphical user interfaces for access).

From a software infrastructure view, Venice is based on a pure service-oriented architecture (SOA) [2] and focuses on openness [9], dependability [5] and security [6]. It is implemented using Web services technology [1]. Its services can be deployed on the Internet – secure and reliable access to the Venice services is managed by the infrastructure. The set of services Venice offers range from service management at runtime, resource and service information and access, collaboration and communication, and utility services for building applications upon. The Venice runtime environment can be used for service development and deployment as well as client development and service access.

Several application and usage scenarios rely on having a file system to reliably and persistently store files. Normally, this is done by using a local file system or mounting a remote file system to a machine where the application is running. But in a service-oriented system, where all services are distributed and work together to form an application, it is impossible to create secure access to one single file system that all services share. Hence a specialized service providing a virtual file system is the best way to seamlessly integrate the functionality of a file system into a service-oriented application. Thus, this file system service can be used by the services that form the application. All security mechanisms of the underlying service framework can be effective. This includes delegation, where permissions can be transferred to other users or services of the system.

2 Related Work

The Amazon Simple Storage Service (Amazon S3) provides a “highly scalable, reliable, fast, inexpensive data storage infrastructure” [8] for storing, accessing and deleting objects from anywhere. Access control lists can be defined for each object. The access protocols are SOAP and REST over HTTP, a BitTorrent protocol interface is also provided.

In the area of Grid Computing and Data Grids, the most prominent solutions for providing access for backend storage systems are dCache [4] and OGSA-DAI [10]. They specialize on large files or data sets. They have a large footprint that makes them difficult to incorporate in a lightweight service Grid, neither on the service nor on the client side.

3 The Venice File System Service

The architectural layout of the File System Service currently being developed is shown in Figure 1. As a Venice service it exposes its functionality through a Web services interface. The interface defines operations for creating, retrieving, uploading and deleting files in the virtual file system. It is also possible to list the files already stored, either per directory or recursively. The meta data of files stored in the virtual file system can also be retrieved. This meta data contains the usual information like read and write permissions or the size of a file, but it additionally provides access control lists that show who might access a file or directory with what permissions. Here, the file system service is directly intertwined with the Venice user and security model.

The functionality that makes the Venice File System Service different from the solutions described in section 2 is the ability to mount other remote file systems into the virtual file system tree without using underlying operating system mechanisms. Several adapters are available that can access directories from FTP or HTTP servers as well as remote file systems that are accessible via SFTP (e.g. the secure FTP daemon that comes with SSH). It is thus possible for a user

to mount the frequently used file systems at the local desktop into the remote virtual file system tree. The implementation uses the Apache Commons VFS library [3] to achieve this. The File System Service then gives coarse-grained access to those remote file systems (in contrast to the fine-grained object-oriented access model of Apache Commons VFS).

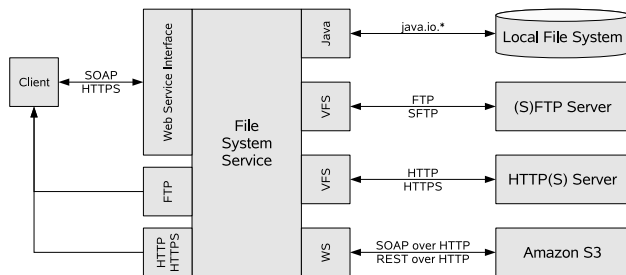


Figure 1. Architectural Layout of the Venice File System Service

Other adapters could connect the File System Service with remote virtual file systems like Amazon’s S3 Web service or dCache sites. From a client’s perspective, there is no difference in using a file from the File System Service’s local file system or a mounted remote file system. The Venice File System Service abstracts from these details and hides them from the clients. It should be noted, that the client has to submit his credentials for accessing a remote file system to the File System Service. This might be a security issue as the client has to trust the File System Service to securely store these credentials.

The File System Service not only stores the files, but it also gives access to those files. A user can retrieve any file stored in the File System Service from virtually anywhere. A straightforward solution would be to retrieve the files through the Web service interface. This solution is applicable for small files due to the nature of XML encoding and SOAP transmissions. Therefore the Venice File System Service also supports other means for accessing stored files. Two standard protocols (FTP and HTTP/HTTPS) will be supported by the service.

4 Conclusion and Future Work

The Venice File System Service provides a virtual file system within the Venice Service Grid that can be used by Venice users or services. It offers the usual file system abilities and additionally mounts remote file systems via (S)FTP and HTTP(S).

The current status of the implementation allows to upload and download files to the service’s local file system. It is also possible to mount remote file systems into the

tree. Accessing those file is currently under development as well as exposing those files to the client side with FTP and HTTP(S). When the implementation is complete, an evaluation will have a look at the performance of the service within the Venice Service Grid.

5 Acknowledgments

This work is supported by the Rheinland-Pfalz cluster of excellence “Dependable Adaptive Systems and Mathematical Modeling” (<http://www.dasmod.de>).

References

- [1] Gustavo Alonso, Fabio Casati, Harumi Kuno, and Vijay Machiraju. *Web Services – Concepts, Architectures and Applications*. Springer Verlag, 2004.
- [2] Thomas Erl. *Service-Oriented Architecture – Concepts, Technology, and Design*. Prentice Hall, 2005.
- [3] The Apache Software Foundation. Commons Virtual File System. <http://jakarta.apache.org/commons/vfs/index.html>.
- [4] Patrick Fuhrmann and Tigran Mkrtchyan. dCache. <http://www.dcache.org/>.
- [5] Bjarne E. Helvik. Perspectives on the dependability of networks and services. *Teletronikk*, pages 27–44, 3 2004.
- [6] Markus Hillenbrand, Joachim Götze, Jochen Müller, and Paul Müller. A Single Sign-On Framework for Web Services-based Distributed Applications. In *Proceedings of the 8th International Conference on Telecommunications ConTEL (Zagreb, Kroatien)*, 6 2005.
- [7] Markus Hillenbrand, Joachim Götze, Ge Zhang, and Paul Müller. A Lightweight Service Grid based on Web Services and Peer-to-Peer. In *Proceedings of Kommunikation in verteilten Systemen KiVS (Berne, Switzerland)*, 2007.
- [8] Amazon.com Inc. Amazon Simple Storage Service (Amazon S3). <http://aws.amazon.com>.
- [9] Andrew S. Tanenbaum and Maarten van Steen. *Distributed Systems – Principles and Paradigms*. Prentice Hall, 2002.
- [10] Open Middleware Infrastructure Institute UK. OGSA-DAI. <http://www.ogsadai.org.uk/>.