

# Community Services: Parallel Scientific Computation Made Easy

Jon B. Weissman, Department of Computer Science  
Associate Professor, Univ. of Minnesota Twin Cities

## Summary

*Building robust high-performance distributed applications is a time-consuming and laborious endeavor due to the heterogeneity and distribution of the underlying resources. Grid services hold great promise in alleviating this burden by allowing such applications to be more easily constructed by presenting a unified view of network resources. To fully realize their potential for providing a stable substrate for such applications, Grid services must cope with uncertain service demand and unpredictable resource availability. The Community Services project is focused on the development of middleware that will address this uncertainty or dynamism at several levels: dynamic service hosting and re-hosting, adaptive resource management, service reliability, and adaptive service classes, all to enable service providers to more rapidly deploy useful services for applications.*

## 1. Introduction

High-performance distributed applications are increasingly multidisciplinary, collaborative, and dynamic. Grid computing which harnesses resources at multiple sites through common access protocols has a great potential to be a future hosting platform for such applications. Recent efforts have focused on standardizing Grid interfaces to enable better interoperability in the form of Grid services. However, Grid services must cope with the uncertainty of service demand and the availability of network resources.

**This is particularly true in laboratories in which services and the resources upon which they are hosted are highly shared.**

If the Grid service provider wishes to provide efficient and robust services, then he/she must develop complex adaptation code as there is little existing infrastructure to utilize. This increases the time-to-deployment which in turns increases the time-to-solution for applications that require such services.

The Community Service project is developing re-usable middleware and

system components that will fill the gap between Grid services and the dynamics of the collaboratory environment (Figure 1). The project is focusing on high-end services which encapsulate parallel and distributed computing, and may access large datasets. Such services may represent kernel codes that have become de-facto standards for production applications. When such codes are transformed into services, several issues arise due to network dynamics: (1) how should resources be allocated to multiple concurrent requests for a high-end service? (2) where should a high-end service be hosted, re-hosted, or replicated? (3) to which service replica should a request be sent? and (4) how to make services appear resilient to failure? Middleware is being developed to address these problems.

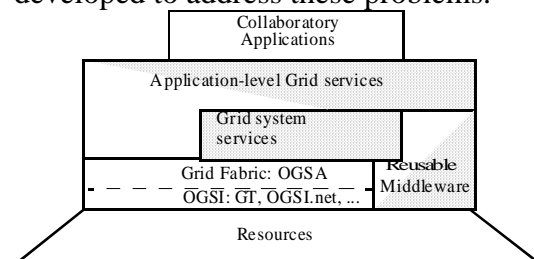


Figure 1. Illustration showing software stack. Project is developing dotted boxes.

## 2. Project Details

The Community Service project is extending Grid services to define an adaptive Grid service – a service that can adapt to current system dynamics. Two classes of adaptive Grid services have been identified: adaptive resource providers (ARP) and adaptive application-level Grid services (AGS). An ARP might be a CPU or storage provider while an AGS could be a parallel solver service, for example. An AGS is deployed or hosted upon one or more ARPs which “lease” resource pools to it (Figure 2). An AGS can be dynamically deployed across the network based on a policy within its deployer module. Once deployed, the service is registered and can be discovered by an application. When an application makes a service request, a front-end (Figure 2, left side) decides which service deployment will handle it (since the service may be replicated). Once the service request reaches a deployed service back-end (Figure 2, right side), it decides which resources it will be given from its allocated pool.

These are all complex decisions which require dynamic performance models and resource management algorithms. A suite of dynamic performance predictors and resource management strategies have been implemented in middleware and made available to services including data-intensive HENP. The middleware is also “smart” and can select the best strategy based on performance feedback. This technique allows the middleware to be more broadly applicable. The middleware also allows new performance predictors and resource management algorithms to be easily added.

A Grid testbed has been created at the University of Minnesota to evaluate system architecture alternatives, develop and test

middleware components, before “roll-out”. An adaptive OGSI implementation based on GT3 has been created as part of the testbed. The project has turned numerous high-end codes into Grid services for experimentation including parallel equation and eigenvalue solvers, genomic comparison, and N-body. The testbed has demonstrated feasibility of the ideas – services can be hosted dynamically and replicated at low-cost and dynamic resource management can achieve significant performance benefits. The next phase of the project is to begin experimentation in the context of large-scale DOE collaboratory projects, e.g. DOE Science Grid, PPDG, Fusion Grid, identifying candidate application codes that can be turned into community services using project middleware.

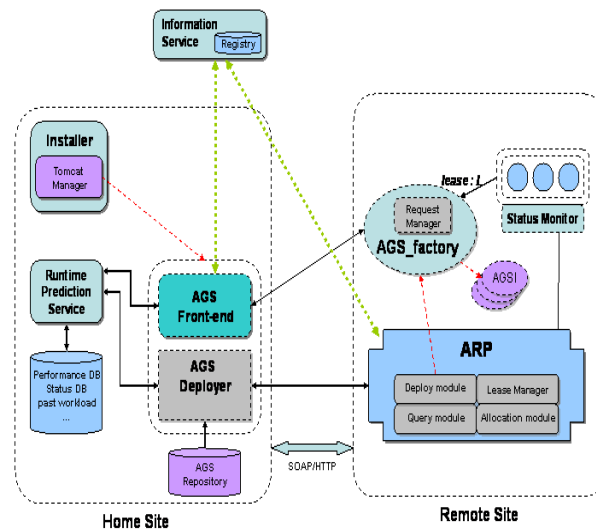


Figure 2. Illustration showing our architecture for dynamic Grid service deployment.

### For further information on this subject contact:

Dr. Jon B. Weissman, PI  
Dept. Comp. Science and Digital Technology Center  
University of Minnesota, Twin-Cities  
URL: <http://community-services.cs.umn.edu>  
Phone: 612-626-0044  
E-mail: [jon@cs.umn.edu](mailto:jon@cs.umn.edu)