

Industry Speak

Leveraging Software Architecture to Reconcile the Promise and Reality of Grid Computing



Nenad Medvidović,
*Associate Professor, Computer Science
Department,
The University of Southern California.*



Chris A. Mattmann,
*Adjunct Assistant Professor, Computer
Science Department,
The University of Southern California.*

Grid computing has been used to solve complex problems in areas such as cancer research, planetary science and earth science. However, its promise is beginning to wane. Nenad Medvidovic and Chris A. Mattmann document the architecture of the grid by undertaking an in-depth study of existing grid technologies.

» [Read](#) | [View profiles](#)



Win in the flat world

Technology Roundtable

September 2009

Leveraging Software Architecture to Reconcile the Promise and Reality of Grid Computing

By Nenad Medvidovic and Chris A. Mattmann of The University of Southern California

Over the past half-century, computing has undergone several transformations that have fundamentally changed the manner in which humans use computers and the nature of problems that can be solved with computers. Grid computing is a recent advance that shows promise of enabling another such transformation. The grid allows virtually any person or organization to solve a variety of complex problems by utilizing the computing resources beyond those of just a small cluster of computers. Today, grids have been used successfully in several domains, including cancer research, planetary science and earth science. A large number of platforms have emerged in quick succession claiming to support various aspects of grid computing. Table 1 summarizes a subset of open-source grid technologies.

However, the early promise of, and excitement about, grid computing has waned recently. At least two new paradigms have supplanted the grid as “the next big thing”: volunteer computing and cloud computing. It can be argued that, from a technical perspective, both of these new paradigms can be considered very close relatives of grid computing and that, in fact, they have been directly inspired and enabled by grid computing. At the same time, this relationship gets conveniently swept under the rug. In fact, as early as 2005 there were indications of a backlash against the grid: The Economist magazine published an article titled, ‘A Grid by Any Other Name’, which indicated that some development organizations were developing grid-like technologies, but insisted on not referring to them as such.

Part of the reaction can be attributed to the incessant hype that surrounded early grid developments. However, as scientists and engineers, we cannot study hype. What we can study is the resulting technology. The early grid literature resulted in several ‘big picture’ publications that tried to establish the underlying principles of the grid: Its ‘anatomy’ and ‘physiology’, which describe the grid’s software architecture, as well as its overarching requirements. An intriguing question, then, is: Are there aspects of the grid’s architecture that can serve as indicators of its rather quick drop-off (if not outright fall) from grace?

Table 1: Several open-source grid technologies

Technology	URL
Alchemi	http://www.alchemi.net/
Apache Hadoop	http://hadoop.apache.org/
Apache HBase	http://hadoop.apache.org/hbase/
Condor	http://www.cs.wisc.edu/condor/
DSpace	http://www.dspace.org/
Ganglia	http://ganglia.info/

Grid-based Light-weight Infrastructure for Data-intensive Environments	http://sunset.usc.edu/~softarch/GLIDE/
Globus 4.0 (GT 4.0)	http://www.globus.org/
Grid Datafarm	http://datafarm.apgrid.org/
Gridbus Broker	http://www.gridbus.org/
JCGrid	http://jcgrid.sourceforge.net/
Object Oriented Data Technology	http://oodt.jpl.nasa.gov/
Pegasus	http://pegasus.isi.edu/
SciFlo	http://sciflo.jpl.nasa.gov/
iRODS	https://www.irods.org/
Sun Grid Engine	http://gridengine.sunsource.net/
Uniform Interface to Computing Resources	http://www.unicore.eu/
Wings	http://www.isi.edu/ikcap/wings/

Kesselman and Foster conducted two seminal studies that tried to underpin and motivate grid technologies. These studies proposed a five-layer reference architecture for the grid, described next from top to bottom:

1. Application houses custom applications, which plug into the common services of an underlying grid infrastructure.
2. Collective aggregates underlying Resource layer services, agglomerating information for a given grid application.
3. Resource encapsulates underlying heterogeneous computing resources (such as files, disks, I/O, etc.) and provides a standard interface for communicating with grid services.
4. Connectivity is responsible for providing security, communication and coordination of access from grid resources to underlying physical resources.
5. Fabric's elements include low-level DBMS, disk I/O, threading and other OS-like resources, available from individual nodes in a grid.

The grid's anatomy disallows 'upcalls', i.e., inter-layer interaction initiated by a lower layer, where a call is an aggregate of explicit invocations from one component to another. The anatomy is ambiguous as to whether interactions can involve non-neighboring layers, although the most widely referenced grid architecture diagram implies that the layers are opaque. Finally, the nature of inter-layer interactions is not elaborated; all interactions are treated as direct (local or remote) procedure calls.

If we look at the as-implemented architectures of the existing grid technologies, however, a number of ambiguities and discrepancies from the reference architecture become apparent. A pilot study involving the architectural recovery (using an existing architectural recovery technique) of the grid technologies listed in Table 1 resulted in the identification of over 250 architectural discrepancies, where none of the grid technologies studied fully adhered to the grid's reference architecture. We explain these discrepancies next and, for illustration, show in Figure 1 structural diagrams of four of the grid technologies' recovered architectures:

1. Empty layers – layers identified in the grid's reference architecture that contained no recovered components; an example is highlighted for Wings in Figure 1a;
2. Skipped layers – components in one layer make calls to components at least two

- layers below or above, highlighted for Pegasus in Figure 1b);
3. Upcalls – calls made from components in a lower (“servicing”) layer to components in a higher (“client”) layer, such as the example for Hadoop highlighted in Figure 1c;
 4. Multi-layer components – components that provided services which, according to the published grid DSSA, belong to two or more layers, e.g., as shown highlighted for iRODS in Figure 1d; and
 5. Orphaned components – components whose exact location in the grid architecture we were unable to determine.

The overall extent to which the existing grid systems’ architectures depart from the reference architecture was surprising. While we may not be able to draw definitive conclusions from our study to date, and while establishing a direct causal relationship between these architectural issues and the shrinking enthusiasm for grid computing would be very difficult at the moment, our study is indicative in several ways.



Figure 1. Representative discrepancies identified in four of the studied grid technologies. At this magnification, the reader is not expected to understand the details of these diagrams, but rather to get a general sense of the four architectures.

[View full image](#)

First, the grid clearly has a different anatomy from the published and widely accepted one. Second, the difficulties of building grids, such as that recently reported by the engineers working on the grid for CERN’s Large Hadron Collider, are likely caused at least in part by the poorly established architectural underpinnings for the grid. Third, any issues associated with the grid are likely to spill over into at least some of the volunteer computing and cloud technologies as some of those technologies are realized on top of existing grid platforms. Finally, and perhaps most importantly, grid computing can and should be treated as an application domain, in which case the grid’s reference architecture becomes a domain-specific software architecture (DSSA).

However, establishing the DSSA, as the progenitors of the grid have done, before a significant amount of experience has been amassed with constructing systems (in this case, grid platforms) in the given domain, is tantamount to putting the cart before the horse and is fraught with peril. We are currently in the process of trying to correct this, by understanding and documenting the grid’s actual architecture through an in-depth study of the existing grid technologies. The sad reality is that we are already getting strong indications from our peers that, while our effort is a worthwhile and even important intellectual exercise, by the time we complete our study its subject - grid computing - may become largely irrelevant.

[« Back](#) to Technology Roundtable Newsletter



Win in the flat world

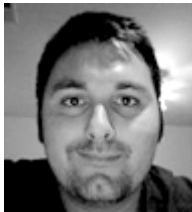
Technology Roundtable

September 2009



Nenad Medvidović is an Associate Professor in the Computer Science Department at The University of Southern California (USC). He is the director of the USC Center for Systems and Software Engineering (CSSE) and the Program Co-Chair of the 2011 International Conference on Software Engineering (ICSE 2011), to be held in Honolulu, Hawaii, May 21-28, 2011.

Medvidović received his Ph.D. degree in 1999 from the Department of Information and Computer Science at UC Irvine. He is a recipient of the National Science Foundation CAREER (2000) and ITR (2003) awards, the Okawa Foundation Research Grant (2005) and the IBM Real-Time Innovation Award (2007). Medvidović's research interests are in the area of architecture-based software development. He is a co-author of a new textbook on software architectures.



Chris A. Mattmann is a senior computer scientist at NASA's Jet Propulsion Laboratory, working on instrument and science data systems on earth science missions and informatics tasks. Mattmann has a joint appointment as an Adjunct Assistant Professor in the University of Southern California (USC) Computer Science Department, where he teaches a course on software architecture to local and remote graduate students over USC's distance education network (DEN).

Mattmann's research interests are primarily software architecture and large-scale data-intensive systems. Mattmann received his Ph.D. degree in computer science from The University of Southern California. He is a member of the IEEE.

« [Back](#) to Technology Roundtable Newsletter

Copyright © 2009 Infosys Technologies Limited
44, Electronics City, Hosur Road, Bangalore - 560 100. Tel: +91 80 28520261