Caveat Emptor: Making Grid Services Dependable from the Client Side

Miron Livny and Doug Thain

Computer Sciences Department
University of Wisconsin-Madison

Abstract

Grid computing relies on fragile partnerships. Clients with hundreds or even thousands of pending service requests must seek out and form temporary alliances with remote servers eager to satisfy them. Yet, despite the high quality and reliability of these servers and their software, unexpected events and behavior are common. Communication networks, power systems, operating systems, middleware and operator intervention all conspire to attack even the most carefully arranged client-server interaction. To survive in such an imperfect world, customers of grid resources must be equipped with resilient client software that tolerates failures while aggressively representing their interests.

Following our tradition of developing technology that harnesses the power of opportunistic resources, the Condor Project is actively engaged in developing the basic mechanisms for building dependable and effective grid computing clients. Guided by our experience and the practical needs of production users in disciplines as diverse as astronomy and sociology, the Project aims to equip users with powerful software that complements the reliability of the servers that they exploit. Our most visible product is the Condor-G job manager. Other research ventures, including the full Condor distributed system, offer valuable lessons in dependable client-side management.

Dependability has been explored in a number of branches of computing, ranging from database systems to programming languages. The hard-earned lessons from these fields are also essential to grid computing. Fundamental concepts such as timeouts, logging, checkpoints, transactions, leases, and atomic operations must be employed and expressed in basic protocols and interfaces for CPU and I/O access. Without these techniques, clients and servers lose track of the other's state, leading to missed opportunities, wasted resources, incorrect results, and unnecessary failures. This principle is espoused in systems such as Condor-G and protocols such as the most recent version of GRAM.

In a Grid environment we must never view failure as a disaster. Rather, failures occur at every level and every interface, and must be expected and structured. No single failure must bring a computation to a halt, nor can any type of failure be retried indefinitely. Jobs may be retracted even from systems deemed reliable when better performance may be found elsewhere. In addition, we must always be careful to determine whether the source of a failure lies in the system or in the job itself. Examples of this principle are found in the DAGMan meta-scheduler and the fault-tolerant shell.

Proceedings of the 2002 Pacific Rim International Symposium on Dependable Computing (PRDC'02)
0-7695-1852-4/02 $17.00 © 2002 IEEE

When un-expected events are common, cleanup must be more aggressive than consumption. If the mere touch of a button allows a user to harness thousands of machines and terabytes of a disk space, we can hardly expect they will expend a greater effort to release what they have allocated. Services must aggressively reclaim allocated resources; otherwise grids will become overwhelmed with unwanted and unending jobs data. Thus services provided by Condor, ranging from the matchmaker to the mobile sandbox, are deliberately designed to quickly and cleanly eject and reclaim resources of failed remote requests.

Each of these lessons has powerfully affected the way we design and implement the Condor software. Although failures are expressed in every component and layer, the client bears the ultimate responsibility and thus must have the most powerful mechanisms at its disposal. Caveat emptor!

Proceedings of the 2002 Pacific Rim International Symposium on Dependable Computing (PRDC'02)
0-7695-1852-4/02 $17.00 © 2002 IEEE