World Wide Failures.

Werner Vogels

Dept. of Computer Science, Cornell University

4105a Upson Hall,
Ithaca, NY 14853

vogels@cs.cornell.edu

Abstract.

The one issue that unites almost all approaches to distributed computing is the need to know whether certain components in the system have failed or are otherwise unavailable. When designing and building systems that need to function at a global scale, failure management needs to be considered a fundamental building block. This paper describes the development of a system-independent failure management service, which allows systems and applications to incorporate accurate detection of failed processes, nodes and networks, without the need for making compromises in their particular design.

Introduction.

With the advent of ubiquitous, world-wide distributed systems, it is becoming clear that the systems that are used today in local-area settings, can not simply be employed in their existing form or trivially converted for wide-area, large-scale operation. Whatever form such systems may take in the future, whether they are replicated databases of hyper-links, distributed objects, view or virtual synchronous groups or agents employing lazy consistency schemes, one of the key problems that needs to be addressed, is that of the detection and handling of faulty components, nodes and networks.

Building distributed systems and applications today is done using a variety of systems ranging from the bare bone protocols interfaces like BSD sockets and the TDI, to RPC based systems such as DCE and to more advanced distributed support systems such as Isis, Horus, Delta-4 and others [1,2,3,11,12]. After years of experience with building these systems and applications, it is clear that failure management is not just a essential tool for group oriented systems, all which have built-in failure handling, but that it is a fundamental service that should be placed among such established basic services as naming, authentication, security, service brokerage and IPC.

This paper reports on an ongoing research effort to abstract the failure handling strategies from a variety of popular distributed systems and to develop a basic failure management service that can be used by any distributed system regardless of the purpose of that system or the techniques used. The strategies employed by this basic service are specifically targeted towards applications that need to operate on a global scale.

Research goals.

To build a successful service the following goals were set:

- design a failure management system that is independent of the distributed systems packages in use and provide failure detection of processes, nodes and networks.
- improve the accuracy of detection of process and node failure through systems support.
- design support for failure detectors to work in large scale systems, while maintaining a high level of accuracy.
- provide support for the detection of partitions in networks.
- build a comprehensive software package that can be easily integrated into various distributed systems packages and applications.

The resulting system is implemented and is under test in a wide-area simulator, in a local setting of a mix of high-speed and traditional networks and in the Internet. A first software release is planned for the autumn of 1996.

History.

External failure detector modules originate in asynchronous distributed systems, where they were introduced to de-couple the mechanism by which failures are detected from the protocols used to tolerate those failures. In [5] Chandra and Toueg successfully show that it is possible to develop consensus algorithms using failure detectors, even if these failure detectors make frequent mistakes in their observations. In [4,7,9] the failure detector work is extended to systems that also take network failure into account.

The major trade-off in designing practical distributed systems based on the theory developed for asynchronous systems is where and how