Distributed Autonomous Fault Treatment in Spread

Hein Meling
hein.meling@uis.no

Joakim L. Gilje
jgilje@jgilje.net

Department of Electrical Engineering and Computer Science
University of Stavanger, Norway

A common technique used to improve the dependability characteristics of systems is to *replicate* critical system components whereby the functions they perform are repeated by multiple replicas. Replicas are often distributed geographically and connected through a network as a means to render the failure of one replica independent of the others. However, the network is also a potential source of failures, as nodes can become temporarily disconnected from each other, introducing an array of new problems. The majority of previous projects have focused on the provision of middleware libraries aimed at simplifying the development of dependable distributed systems, whereas the pivotal deployment and operational aspects of such systems have received very little attention. In traditional fault tolerance frameworks, one relies on the system administrator being able to replace failed replicas before they have all been exhausted, which would cause a system failure.

This paper presents a novel architecture for *distributed* autonomous replication management (DARM), aimed at improving the dependability characteristics of systems through a self-managed fault treatment mechanism that is adaptive to network dynamics and changing requirements. Consequently, the architecture improves the deployment and operational aspect of systems, where the gain in terms of improved dependability is likely to be the greatest, and also reduces the human interactions needed. The architecture builds on our previous experience [3] with developing a prototype that extends the Jgroup [4] object group system with fault treatment capabilities. The new architecture is implemented on top of the Spread group communication toolkit [1] and relies on a distributed approach for replica distribution (placement), thereby eliminating the need for a centralized replication management infrastructure used in our previous work [3], and also in all other related works.

DARM is implemented by means of an interception layer (called libdarm) between Spread and the application to handle system reconfiguration in case of failures. Furthermore, a separate factory component is used on each node in the conjunction with the Spread daemon to offer applications support for installing new replicas to replace failed ones. The libdarm library is designed as a trivial replacement for libspread allowing Spread applications to become self-healing with respect to replica and network failures. The following replica distribution policy is implemented by libdarm, allowing seamless deployment and recovery from failures: (i) find a network segment without a replica of the given type, (ii) keep the number of replicas in each segment evenly distributed on the nodes, and (iii) find the least loaded node that is not running a replica of the given type. The required policy configuration parameters are provided by the application at runtime.

Additional details about the approach, as well as some initial measurements can be found in [2].

Referanser

[1] Y. Amir, C. Danilov, and J. Stanton. A Low Latency, Loss Tolerant Architecture and Protocol for Wide Area Group Communication. In *Proc. of the Int. Conf. on Dependable Systems and Networks*, New York, June 2000.

[2] J. L. Gilje. Autonomous fault treatment in the spread group communication system. Master's thesis, Unversity of Stavanger, June 2007.

[3] H. Meling. *Adaptive Middleware Support and Autonomous Fault Treatment: Architectural Design, Prototyping and Experimental Evaluation*. PhD thesis, Norwegian University of Science and Technology, Department of Telematics, May 2006.

[4] A. Montresor. *System Support for Programming Object-Oriented Dependable Applications in Partitionable Systems*. PhD thesis, Dept. of Computer Science, University of Bologna, Feb. 2000.