DeDiSys — An European Union aided Research Project**

M. Jandl, A. Szep, R. Smeikal
Vienna University of Technology
Institute of Computer Technology
Gusshausstrasse 27-29/384,
A 1040 Vienna, Austria
+43 1 58801 38448
{jandl|szep|smeikal}@ict.tuwien.ac.at

K. M. Goeschka
Vienna University of Technology
Institute of Information Systems
Argentinierstrasse 8/184-1,
A 1040 Vienna, Austria
+43 1 58801 18412
Karl.Goeschka@tuwien.ac.at

Abstract

Distributed software systems are the basis for many innovative applications (e.g. pervasive computing, telecommunication services, and control engineering). Many business cases focusing on specialized know-how in the global market of software components, services, and infrastructures built upon these systems can be identified. The key for achieving scalable and maintainable distributed systems is dependability, because otherwise the complexity of distribution would leave the system uncontrollable. Hence, our approach aims at a concept for optimizing dependability. Similar to other approaches we use replication as means to provide transparent fault-tolerance and persistence, but we especially focus on increasing availability by relaxing data integrity by using a mixture of asynchronous and synchronous replication techniques. This trade-off allows for a configurable and application-specific optimum of availability, possibly even controlled during runtime. **

1 Project Motivation

Today, dependability is mainly taken into account for safety critical applications, but it is also required in the fields of telecommunication, control engineering, and ubiquitous computing. Unfortunately, the inherently complex distributed systems are often unmanageable, unless fault-tolerance is already built into coherent system parts and not scattered over the system. There exists no "fault-tolerance"-module that can be added later on—dependability is an aspect of every part of the system. Failures can be classified as site failures (a particular site is not working) and network failures (in particular network partitions, where a group of sites is unreachable but still operating). Our research aims at a concept for optimizing dependability in distributed software systems by relaxing data integrity (consistency) for availability, since for a class of applications availability is more critical than volatile inconsistencies. The following real-life applications motivate this trading and envision the exploitation potential for further improvements.

The **Distributed Telecommunication Management System** (DTMS) is a monitoring and control system for a safety-critical Voice Communication System (VCS) in the field of air traffic control (ATC). Multiple DTMS servers monitor and control their associated VCSs which store parameter data in objects. By sacrificing consistency in case of failures, a client can be provided with an expected level of availability. The planned models should decide dynamically, basing on their configuration, how much consistency is to be traded for availability.

Flight Objects: The European Union prioritizes the creation of the Single European Sky for ATC [1]. "Flight Objects" are a form of distributed intelligence which can leverage this vision—they are used to model all relevant flight data for ATC. Not every user requires a consistent view on all data, some users are satisfied with slightly outdated *but immediately available* data. Together with replication, such requirements allow for higher availability as compared to systems with strict consistency requirements.

**The European Union supports this research project under Framework Programme 6 within the IST priority – Project title: "Dependable Distributed Systems", Acronym: "DeDiSys", Contract No.: 004152.

2 Solution Approach

Strong consistency is often deployed but not always desirable, because it also implies strong limitations of availability. Generally, the trade-off between replication availability and consistency cannot be configured in such systems. Therefore, our solution approach allows for fine-grained tuning of this trade-off. The key idea is to use a mixture of asynchronous and synchronous replication techniques, whereby asynchronous replication is used to replicate persistent object-states, while operating on objects is implemented synchronously. Asynchronous propagation of object data prepares for degraded scenarios. Therefore, the central concept is the trade-off between availability and consistency and a mechanism to communicate current limitations to a client and, even more importantly, allow the client to react selectively, is required.

3 Contribution and Expected Results

Firstly, the above mentioned trade-off between availability and consistency shall be measurable and possibly even configurable during runtime. Secondly, as location and access transparency are the basis for most other relevant middleware properties, one possible system model is the Fault-Tolerance Naming Service (FTNS), which comprises the fault-tolerant mapping from object identity to reference, which is potentially different on every node because it depends on the view of the current failure scenario at a particular node and on the replication algorithm. Besides this, consistency information must also be provided. For example, a client trying to access a certain object could obtain a list of locations and consistency levels from the FTNS, and accesses the location that has the data at an acceptable consistency level.

While the current DTMS implementation only allows for a rather coarse-grained trading between availability and constraint consistency, we expect as long-term goal to move continuously between increasing availability for consistency and vice versa, if the system is degraded.

4 State of the Art and Related Work

Complex applications usually demand both transparent distribution and persistence, and proven frameworks (EJB, COM+/.NET, CORBA - all with their associated services) exist. However, these frameworks provide fault-tolerance only in connection with strong data consistency. Clustering is a common solution for high availability in server farms, usually deployed for load-balancing in a local area network where network partitions are extremely unlikely. We on the other hand explicitly address the problem of accessing objects in different network partitions.

The TACT project [5] is among the mature research of the trade-off between availability and replica consistency and further approaches, using group communication and object groups, exist [3], [4], [6], [7], [2].

However, all of the above mentioned solutions either deploy strong consistency or leave replica management entirely to the application. Generally, the trade-off between replica availability and consistency cannot be configured in such systems. Our approach in contrast allows for fine-grained tuning of this trade-off even during runtime.

References

[1] European Commission. *"White Paper – European transport policy for 2010: time to decide."*, Luxembourg, 2001, ISBN 92-894-0341-1.

[2] K.P. Birman. *"The process group approach to reliable distributed computing"*, Communication of ACM, 36(12):37-53, December 1993.

[3] D. Malkhi et al. *"Persistent objects in the fleet system."*, In Proceedings of the 2nd DARPA Information Survivability Conference and Exposition (DISCEX II), June 2001.

[4] L. E. Moser et al. *"The eternal system: An architecture for enterprise applications."*, In Proceedings of the International Enterprise Distributed Object Computing Conference EDOC 1999, pages 214-222, September 1999.

[5] H. Yu and A. Vahdat. *"Design and evaluation of a conit-based continuous consistency model for replicated services."*, ACM Transactions on Computer Systems, 20(3):239-282, August 2002.

[6] R. van Renesse et al. *"Horus: a flexible group communication system."*, Communication of ACM, 39(4):76-83, April 1996.

[7] K. Birman et al. *"The horus and ensemble projects: Accomplishments and limitations."*, In Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX 00), January 2000.