Post-Client/Server Coordination Tools

eva Kühn and Georg Nozicka*

University of Technology Vienna
Institute of Computer Languages
Argentinierstraße 8, 1040 Vienna, Austria
{eva,nozicka}@complang.tuwien.ac.at

Abstract. The exploitation of new application possibilities, like collaboration and cooperation, offered by distributed systems requires advanced coordination support. Traditional tools are based on the message passing paradigm and lead to asymmetric client/server application architectures. The other – conceptually superior – paradigm uses a virtual shared memory. The development of distributed programs is easier in the latter model and leads to elegant solutions that meet well the new possibilities. We term software support that follows this second approach *post-client/server* tools. CoKe *(Coordination Kernel)* is a new middleware layer of this new generation. It particularly eases the development of fault-tolerant, distributed applications.
We discuss, why coordinative data structures (on *virtual shared objects*) provide more advantages than the traditional method invocation model (on *distributed objects*).

1 Introduction

Computer networks offer new application scenarios that cannot be realized on single workstations. Resources and services may be distributed because of political, organizational, etc. reasons which make a centralization impossible. The parallelism gained through the simultaneous processing of subtasks at distributed sites may lead to a better system throughput. Architectures comprising workstations connected to Intranets, distributed Intranets or the Internet offer a better price/performance behavior for coarse grained, commercial applications than parallel hardware. The market for the latter one being primarily massively parallel and highly scientific applications.

Each connected workstation contributes to the common *global distributed architecture*, which makes the hardware scale-up easy to plan and to manage. To participate in the pool of common hardware, one has simply to connect a new workstation or a network terminal.

New application domains offered by the distributed hardware are for example:

* Current address of the author: IBM Austria Ltd., Lassallestraße 1, 1020 Vienna, Austria, nozicka@vnet.ibm.com.

- **Electronical Supermarket.** Many components can be found in the network that need not be reinvented. This idea exceeds the traditional notion of software reuse, which basically refers to existing program libraries. In the network, data and services are offered that need not resp. cannot be transferred to the own site, but rather are to be used on the site where they are installed and maintained. Programming will move towards organization of distributed resources [16].
  Sophisticated accounting and security systems are needed for this supermarket, the implementation of which in turn requires advanced software tools to guarantee their reliability.
- **Workflow Management.** A workflow management system describes units of work and specifies the data and control flow between different activities, including constraints and dependencies. Workflow serves to reengineer business and information processes with the goal to reduce the cost of doing business [2].
  A workflow management system involves many distributed resources and users that contribute to the global task in several roles.
- **Multi Database Systems.** A multidatabase systems integrates several, distributed autonomous database systems, that are pre-existing (so-called legacy systems). The objective is to offer a unique and homogeneous view on the different data, models, and representations, while keeping local system's autonomy [1, 11].
  Besides the semantic integration issues, the heterogeneous transaction processing is a main aspect of a multi database system. Communication and site failures must not destroy the global transaction's semantics.
- **CSCW.** The cooperation of teams on larger tasks via electronic communication facilities adds more flexibility for employees, including the possibility of teleworking.
  However, computer supported cooperative work in the network requires complex transaction and synchronization mechanisms. For example, the early commit of subtransactions that allows intermediate results to become visible, before the global transaction commits, is a new requirement here. Advanced transaction models like the Flex Transaction Model [3] have been designed for these meets.

Alas, these issues pose a lot of new challenges on the application developer. Obviously, the new technical possibilities and requirements cause distributed applications to be more complex and difficult than software for a single computer; take only into consideration how difficult it is to verify the correctness of programs if deadlocks must be avoided, heterogeneity of the different systems, data types and models must be hidden, reliability of communication including message loss or duplication has to be considered, security and accounting has to be established to avoid unauthorized accesses, concurrent and parallel activities must be synchronized, etc.

Besides these extra burdens, however, distributed architectures allow for software improvement through exploitation of the distribution. As is known, replication of data and services can improve availability and reliability, and also