Composable QoS-Based
Distributed Resource Management

Nalini Venkatasubramanian and Gul Agha
University of Illinios at Urbana-Champaign
{nalini,agha}@cs.uiuc.edu
Carolyn Talcott
Stanford University
clt@cs.stanford.edu

1 Motivation

System-wide properties such as reliability, availability, security, and responsiveness are realized by resource management policies. Current techniques for distributed programming merge application code and code implementing resource management policies, making it difficult to change one without impacting the other. Separate specification of resource management policies allows dynamic modification of the protocols needed to meet changing system conditions or requirements. In order to facilitate such separation, one needs composition mechanisms that allow customization of meta-level services, and their dynamic composition with applications.

Different service specifications will require different protocols to be implemented for services such as placement, scheduling, replication, clock synchronization, migration etc. The ability to use component-based software requires that different service specifications are met without changing code for an application's components. This in turn requires that the components implementing the protocols that are used to satisfy service specifications do not interfere in ways that prevent each other from carrying out their actions.

2 Two-Level Architecture

A two-level model of distributed computation, called TLAM, provides for the separation of application and system requirements. TLAM is the basis for developing a semantic framework that supports dynamic customizability and separation of concerns in designing and reasoning about components of *open distributed systems* (ODS). The TLAM meta-architecture framework distinguishes between two kinds of subsystems—application subsystems with application objects and meta-level subsystems with meta-level objects. Meta-level abstractions and facilities constitute the building blocks of a *meta-level distribution infrastructure* for managing open distributed systems. One or more meta-level controllers define protocols and mechanisms and can customize the placement, scheduling, and management of actors in a distributed system. This adds a new dimension of control and flexibility to system management.

We envision a group of core services that can be composed in various ways to support protocols acting at a meta-level. Use of such core services simplifies reasoning about non-interference properties. Consider a distributed Multimedia system. Media data management must decide where to place the data actors in such a system, while media request management must figure out how to schedule actors to guarantee the properties (availability, timeliness, etc.) required by a service request. These functions in turn require a number of services. For example:

Synchronization service schedules message service in order to maintain a uniform notion of time among actors in the session that are distributed across multiple nodes.

Replication service replicates data and request actors (to support availability, load-balancing and fault tolerance).

Dereplication service can dereplicate data or request actors.

Migration service migrates data or requests (to support load balancing, availability and locality).

Each of these services in turn can be based on one of the two core services—remote creation and distributed snapshot. The organization of meta-level services provided by a QoS broker based on these two core services is illustrated in Figure 1.

3 Discussion

The dynamic nature of applications such as those of multimedia under varying network conditions, request traffic, etc. imply that resource management

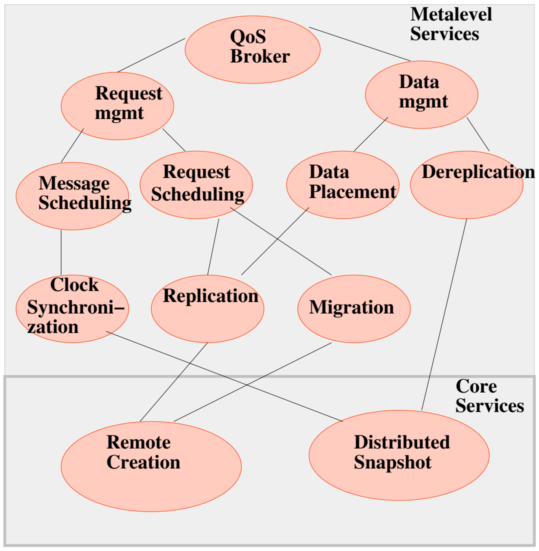

Figure 1: Detailed Architecture of the Meta-level System

policies must be dynamic and customizable. Current mechanisms, which allow arbitrary objects to be plugged together, are not sufficient to capture the richness of interactions between resource managers and application components. For example, they do not allow customization of execution protocols for scheduling, replication, etc. This implies that the components must be redefined to incorporate the different protocols representing such interaction. We believe that a cleanly defined meta-architecture which supports customization and composition of such protocols is needed to support the flexible use of component based software.

References

[1] S. Ren, N. Venkatasubramania, and G. Agha. Formalizing multimedia QoS constraints using actors. In H. Bowman and J. Derrick, editors, *Second IFIP International Conference on Formal Methods for Open Object-based Distributed Systems, 1997*. Chapmn & Hall, 1997.

[2] N. Venkatasubramanian and C. L. Talcott. Reasoning about Meta Level Activities in Open Distributed Systems. In *Principles of Distributed Computation*. ACM, 1995.