Metacomputing In Large Asynchronous Networks

Calypso, Chime, Charlotte: Fault-tolerant Execution

ResourceBroker: A Responsive Resource Controller

The MILAN software architecture consists of three application-level layers interacting with three system-level layers. The application layers provide (from top to bottom) a platform-independent specification of the computation, application-aware middleware which enables the computation to adapt to changing resource availability profiles, and a QoS agent which negotiates an appropriate level of resource allocation during the program’s lifetime. The system layers provide (from bottom to top) a resource controller which keeps track of resource availability, an application-independent virtual-machine interface layer that enables user computations to execute across heterogeneous resources in a fault-tolerant and efficient fashion, and a QoS arbitrator which responds to QoS agent requests with a level of resource allocation that can satisfy application QoS requirements.

Different layers of the MILAN architecture are active at different times in the program’s lifetime. The above figure shows a typical progression: (1) the resource controller layer adds resources into the system, (2) on program arrival, the user program, the application QoS agent, and the system QoS arbitrator participate to determine a resource allocation for the program, (3) the user program and application-aware middleware stages adapt to the available resources, (4) a change of resource availability is detected by the resource monitor which triggers a renegotiation of resources between the application and the system (step 2).

Access Proxies

COTS systems via protocol customization. The framework first builds a model of program interaction using compile-time analyses and run-time profiling, and then, utilizing this model, derives code fragments (access proxies) which construct custom protocols from flexible run-time primitives.