QoS in the Off++ kernel

Extended Abstract

Francisco J Ballesteros **Fabio Kon** **Roy Campbell**

nemo@gsyc.escet.urjc.es {f-kon,roy}@cs.uiuc.edu

In the highly-dynamic, heterogeneous environments of the future, applications will use various different kinds of resources found in local networks. For example, during a meeting, an application can transfer audio of a recorded lecture from a network file server to a sound card connected to the meeting room speakers. Later, when users leave the meeting room, they may choose to have the audio data sent to their cellular phones so they can continue listening to the recorded lecture.

The time for ubiquitous computing envisioned by Mark Weiser and promoted more recently by Sun's Jini technology is approaching very rapidly. In the near future, our networks will be connecting a plethora of dumb devices like TVs, stereos, VCRs, keyboards, sound cards, disks, etc. Furthermore, the network will be in continuous change; new devices will come and go all the time.

Current operating systems are not well prepared to manage this highly dynamic and heterogeneous environment with proper provision for Quality of Service. New approaches to deal with these kind of systems, like Jini, have not yet addressed the problem of dynamic resource management and Quality of Service.

To address these problems, we are developing Off++ [http://gsyc.escet.urjc.es/off], an object-oriented kernel designed to export heterogeneous networked resources to user applications and to let applications adapt to changes in resource availability. Off++ is the kernel of *2K* [http://choices.cs.uiuc-.edu/2k], a distributed operating system designed to support users and applications in dynamic, heterogeneous environments. Although Off++ runs on the bare hardware, its services can also be emulated on top of Unix and Windows systems.

Off++ exports networked resources to authenticated users in the form of distributed objects. All abstractions and system resources implement a single interface, called "Box", that allows applications to inspect and modify dynamic resource state. Applications can allocate, use, and deallocate Boxes in any networked device.

Off++ provides the basic mechanisms required to build support for QoS guarantees in a distributed system by allowing user applications, or the middleware, to reserve low-level network resources. This is achieved through *architectural-awareness*: middleware and application code can navigate through system components and inspect component properties using the Box interface. For example, one can navigate a node to locate memory banks, then ask for the value of the "DMA-capable" property, and locate chunks of memory supporting Direct Memory Access. Off++ nodes can guarantee the allocation of a given resource (including processor time, physical memory, network, and so on).

In few words, to offer a service with guaranteed QoS, an application can: (1) navigate through the set of resources in the network; (2) for each resource, inspect the values of some of its properties (e.g. "size", "availability", "bandwidth", etc.); and (3) reserve and allocate the required resources. All that using a single interface. Furthermore, the system can be instructed to notify the application about changes in resource properties, leading the way to application adaptations.

We believe that, in comparison to existing systems, our approach provides a better fine-grain control of networked resources in heterogeneous environments. We plan to port the soft real time scheduler DSRT [http://monet.cs.uiuc.edu/dsrt2/] to Off++ to demonstrate this, and to see if our approach will allow QoS-aware network applications and middleware to be built more easily.