Application of Operating System Concepts to Coordination in Pervasive Sensing and Computing Systems

Jesse M. Davis, Joseph B. Evans
Information & Telecommunications Technology Center
University of Kansas
Lawrence, KS 66045

Benjamin J. Ewy, Larry M. Sanders
Ambient Computing, Inc.
3514 Clinton Parkway, Suite A257
Lawrence, KS 66047

Abstract

An ambient computing environment coordinates a variety of computing and network-enabled devices to present a seamless, customizable and invisible interface to the user. The idea of the meta-operating system applies the qualities of traditional operating systems to this environment. This allows devices and applications to be created and integrated much like they are in traditional operating systems. The meta-operating system also allows personalization, permissions, and location-specificity to be offered in a natural way.

Introduction

Traditional approaches to consolidate and control the increasingly large number of information-based devices in military, government, business, and residential environments have not provided an integrated infrastructure that easily supports the development of applications that make use of network-embedded sensors and controllable devices. This situation is analogous to a traditional computer in which there is no operating system to coordinate the mouse, keyboard, display, network interface, and many other components that make the computer a usable tool - such a situation stifles application development and is simply intolerable.

Applying operating system qualities to a distributed, network-embedded device architecture creates a system capable of enabling many new services by providing a common way to integrate and coordinate devices and to create applications. Personalization, permissions, and location-specificity can then be incorporated into this architecture in a similar fashion to operating systems. This provides developers with a consistent and secure infrastructure that can be used to facilitate applications that receive inputs from, and send output to, a variety of network-embedded devices as well as existing applications on network-connected computers. It also provides users with a more information-rich environment, tailored to their requirements, but without imposing the user overload associated with individual tweaking of collections of gadgets.

Applications

These concepts can be applied to a variety of applications, including military and government systems.

The meta-operating system concept might be applied to tactical situations in which there are a large number of sensors deployed, only some of which are useful at a given time, or for which particular personnel are appropriately cleared. In this scenario, the meta-operating system, with components running on sensors and communications gateways, provides the infrastructure to write an application which allows certain personnel to see particular sensors based upon their clearances and perhaps location. For example, a simple application might allow certain soldiers access to data from motion and audio sensors, but only intelligence officers access to data from communications sensors.

Meta-operating systems can provide the infrastructure for building more complex applications that span multiple sensors and platforms. An example scenario might be the integration of chemical, biological, and radiographic sensors in homeland security applications. The use of a meta-operating system allows rapid development and deployment of new event collection and filtering applications that can exploit new sensors and processing algorithms, while at the same time providing the appropriate customized views into the system by developers, investigators probing for particular incident types, and general security forces monitoring for attacks.

Finally, ambient computing systems might be utilized in more mundane environments such as the residence, in order to enable the long-promised "smart home" scenario.

Related Work

The meta-operating system concept builds upon work in a variety of fields.

Supporting infrastructure technologies for ambient computing systems include wireless networking, e.g. 802.11, for device connectivity, XML for message and data exchange, and directory services such as LDAP for structured information storage and retrieval. Service location protocols, such as SLP, allow portions of the system to discover and advertise services to other parts of the system. Existing user interface technologies such as the Web, instant messaging, WAP, and related protocols provide different ways to communicate with users. Speech recognition and synthesis tools also provide for a natural interface to the system in some circumstances.

Other systems for discovering and controlling network devices such as UPnP provide part of the needed functionality, but not a complete solution. In particular, UPnP provides discovery, communications, and control capabilities; it is analogous to a PnP PCI bus, but not the rest of the infrastructure that exists beneath applications. The meta-operating system approach incorporates UPnP and other service discovery protocols as part of a framework that application developers and end-users can exploit.

The meta-operating system concept is also different from distributed computing efforts such as the Grid and the Globus infrastructure. Traditional distributed computing is oriented around partitioning a single computational problem or set of problems into pieces and solving those bits in parallel to hasten the completion of an overall solution. The recent work in this field is enabling larger collections of resources to be accessed and used by large communities, with some nascent support for integration of instrumentation. Meta-operating systems are addressing a different problem, that of integration of large numbers of distributed input/output resources with computation so that environments can be more information-driven.

Recent work in sensor webs, including other publish-subscribe systems, are also addressing the need to network and process large amounts of information, but typically without the architectural framework that is provided by the meta-operating system which enables a consistent and familiar way for rapidly developing applications.

Architectural Overview

The MetaOS we have developed is a particular implementation of the meta-operating system concept. The MetaOS is based on a publish-subscribe architecture. The "kernel", or hub, receives messages from the clients, or edges, on behalf of devices managed by the edges. These messages trigger notifications and operations on the hub, which change the MetaOS' view of the environment. Messages are then sent back to the devices to make these changes in the environment. The MetaOS architecture can support multiple hubs in order to implement multiple domains of control for large collections of devices, and redundant hubs for reliability and resilience.

Edges and hubs communicate using TCP/IP as the transport layer. XML messages define a set of operations that may be performed on the hub, as well as describe a format by which devices can send information about the environment and receive instructions on how to interact with it. This common format allows a myriad of devices to interact with the MetaOS through a standard API, removing the need for device-specific logic to be incorporated into the system. This is based on the device driver model used in modern operating systems, where the core logic of the system is accessed through a well-defined interface.

These messages include:

- Registration messages for devices, notifications, operations to take upon receiving notifications, and macros.
- Notification messages sent from devices when external stimuli are noticed.
- Operation messages sent from the hub to instruct devices on how to respond to the given stimuli.
- Informational messages that query a data store to retrieve customization parameters.

As in a traditional operating system, applications can be written that interact with the system devices through the hub. Process management for applications is also provided.

An outside data store is used to store system and device preferences that are customized by the user. An abstract interface allows applications to access any type of data store, from LDAP directory services to transactional SQL databases such as Oracle. Applications can use these preferences to tailor their behavior and that of devices to the user's needs.

Users can then use the system by validating themselves using many different ways, such as a Web login, voice recognition system, smart card, fingerprint reader or other biometrics. During the user's session, many interfaces to the system can be used, including a web application and voice commands.

Summary

A MetaOS implementation has been completed, with portions of the system in Java, C, and Perl. The system has been tested in a number of scenarios utilizing a variety of sensors, actuators, and applications, including input devices such as network-attached cameras, motion sensors, temperature sensors, control devices such as powerline controllers, output devices such as traditional displays, network-capable PDAs, PCS phones, pagers, and applications such as web browsers, messaging agents, virtual desktops, audio players, and display software. It is expected that this infrastructure will be applied to the specific application domains that can best utilize network-embedded device integration, for example sensor networks for homeland defense.