By Ken Klingenstein

The Middleware Years of the Internet

As breathtaking as the emergence of the Internet has been so far, the invention is far from done. A second layer of global interconnectivity is now being engineered and deployed on top of the physical connectivity of computers. This "middleware" layer focuses on the interconnection of people and services rather than on networked computers.

There are several versions of this middleware vision. Imagine that you have a set of physical tokens (smartcards, USB dongles, or perhaps some of the new Java buttons) that you plug into the information appliance at hand to temporarily customize it to your electronic worldview. Middleware represents the network mesh of services and protocols that hold and deliver your e-persona to the requisite device. That appliance may be your desktop, where you unlock your encryption tools. It may be in a student computing lab, where bookmarks, e-mail aliases, encryption tools, and preferences are all downloaded, on application of the hardware token, for individual student use. The appliance may be an airport kiosk, where you could send your e-mail without having to fumble to remember the full addresses of long-ago aliased colleagues. It may be a rental car, which will soon be equipped with USB ports so that after a small pause that reflects the negotiations of some protocol still to be written, the car radio buttons are set to your musical preferences.

The definition of "middleware" may depend on one's point of view, but it typically applies to sets of tools and data that help applications to use networked resources. Some services, such as authentication and directories, are common and have come to be known as core middleware. Other middleware services, such as coscheduling of networked resources, secure multicast, object brokering, and messaging, are the particular interests of specific communities such as scientific researchers or business systems vendors. The following definition reflects this breadth of meaning: "Middleware is the intersection of the stuff that network engineers don't want to do with the stuff that applications developers don't want to do."

The most obvious driver for this middleware layer is the burgeoning number of applications that a typical user now deploys. Many applications require us to fill in e-mail addresses, or pick preferences, or specify personal title descriptions. It would be useful to store such information in a single location for multiple applications to access. Picking a single, integrated suite of applications is an appealing alternative, but it is unlikely that such a suite really exists. Moreover, the requirements for mobility of persona (as well as interactions with information appliances of all ilk) suggest an open and broad implementation.

Other drivers for middleware are also important. As electronic privacy issues become critical, we will want to have our encryption keys at hand. We will also want to shield our preferences. Placing them into a single, carefully designed storage mesh is safer than placing them, often repeatedly, into a host of applications and appliances. There are compelling niche drivers as well. For example, advanced scientific applications need to pass their preferences between servers and instruments worldwide in real time. And the full version of e-commerce will require individual authorization and security tools far beyond the simple secure channels in use today.

Why Is This Hard?

The challenge of deploying core middleware does not seem technically difficult or costly, particularly when compared with the implementation of physical connectivity. There is invention, to be sure, but in the end it is just software and standards. The harder aspects of deploying middleware involve policies and politics. In connecting people to the network, we are automating some of the most nonlinear parts of our world.

Examples of the politics are, unfortunately, legion. For example, as long as an institution's student information system and the human resource system are separate, it will be difficult to notice if a student-worker's physical mail address is different in the two systems. If we enable those systems off a common directory, the inconsistent addresses will be immediately visible, likely precipitating a long and contentious discussion over which system has the better data to trust. Similarly the policies, or lack thereof, around key identifiers, such as e-mail addresses and login names, are exposed. Who can have the identifier? Can that value be reused for another individual? Can a permanent identifier be changed? For which applications is a particular identifier most appropriate? Such inconsistencies and gaps are commonplace at most colleges and universities.

In addition, the number of "moving parts" in middleware is quite high, and multiple options exist for serving most functional requirements. The approach at an individual institution depends on a number of local constraints and factors, most notably the embedded bases of technology and organization. Unlike physical network infrastructure, which is now deployed in somewhat "cookie cutter" fashion in higher education, middleware depends on institutional systems, organization, and procedures, making its deployment more of a design process than a construction project. Indeed middleware is the realm of IT architects—specialists who use common building blocks (which in this case make up a dazzling list of acronyms like LDAP, PKI, SSL, GSSAPI, etc.) to create services that fit the "neighborhood and needs" of students and staff.

Also, the deployment costs typically will be spent less in physical plant and capital and more in time: time to discover the nest of *ad hoc* policies and controls of the existing information sources, time to ferret out the redundancies and inconsistencies when different departments manage similar information, time to write the interfaces between the legacy systems, and time, endless time, to build consensus and agreement in order to actually implement the relatively straightforward technologies.

If this degree of integration is so difficult, then why do it at all? Just like physical facilities and Internet Protocol (IP) networks, a consistent and ubiquitous core middleware plant on campus will support the confluence of applications that will ride on top of these services. A coherent infrastructure will improve reliability, reduce data discrepancies, ease maintenance, contain costs, and encourage the same externalities as exploited at the IP layer.

What Should an Institution Be Doing?

Within core middleware there are areas—such as identifiers, authentication, and directories—where the technology is well in hand and the challenges lie in institutional deployment efforts. There are also some areas—such as PKI (Public Key Infrastructure) and authorizations—where we still have much to learn and invent. So there is a mix of initiatives and experiments that colleges and universities should be working on:

1. *Get the institutional identifier and namespace in order.* The Internet2 middleware Web site (<http://middleware.internet2.edu>) defines a mapping process that a number of universities have used to work through the substantial issues and policy definitions within this task. Your institution will likely need to create one or two new identifiers to accommodate the variety of electronic requirements. It is not that there is a "right" set of policies for the use of identifiers; the challenge is in understanding, and perhaps rationalizing, the policies.
2. *Clarify institutional authentication strategies and prepare for some improvements.* Authentication covers both the initial identification of a person and the electronic and physical processes used to confirm that identity in subsequent interactions. Technologies need to be deployed, for example, to eliminate clear-text passwords, to reduce the number of logins required, and to develop alternative authentication modes that can carry additional information. Policies are needed to manage user protection of passwords, to protect and audit the authentication systems themselves, and to establish levels of assurance that individuals are who they say they are at first contact.
3. *Start building directories, while paying attention to processes and emerging consensus approaches.* The enterprise directory service, likely consisting of several specific subcomponents, is the core of core middleware. There are numerous places where technical design must be influenced by political issues, and there are instances where technical requirements will necessitate institutional adaptations. But a basic directory deployed by the central IT organization can provide important experiences and can also quickly enable improvements in e-mail services, account management, and Web access controls. A number of national standards that are emerging in the design and deployment of campus directory services will prove helpful.
4. *Prepare for PKI.* Building directories is a key step in this process, but other groundwork is needed. This is still a land of invention, but institutions should be creating pilot projects to begin to understand the technical issues. Deploying a PKI will require the participation of legal counsel, applications developers, systems managers, and user support services. Get the partnerships formed and informed on the issues. When PKI finally does come, it may come hard and fast.

There is consequential work at hand. A critical new fabric is being created, one that clearly brings deep human issues into the design of technology. Personal and institutional information will be woven together into a global mesh. This should be done with great care, but it should be done.

Ken Klingenstein is Director of the Internet2 Middleware Initiative and Chief Technologist at the University of Colorado at Boulder.