application needing to invoke a capability or service in more than one context. If we are passing actual data between systems, the application developers become responsible for knowing about the access control issues when crossing application boundaries. If we instead pass a reference to the data, the initial application is no longer responsible, and we can keep the information-driven centralized access control working for us. Many existing SOA systems restrict access to services based on identity or role, but they rarely support restrictions to the specific data that passes through these services. This limitation is part of what makes conventional web services simultaneously confusing and insufficiently secure. Access policies should be applied to behavior and data within a context, but without being able to name the data and contain the context, that becomes very difficult.

When people first begin exploring resource-oriented architectures, they get concerned about exposing sensitive information through links. Somehow, returning blobs of data behind opaque queries seems more secure. They have difficulty separating the act of identifying and resolving something from the context in which it is done. That context contains sufficient information to decide whether or not to produce the information for a particular user. It is orthogonal to the request itself and will be met by the extant authentication and authorization systems in place for an organization. Anything from HTTP Basic Auth, to IBM's Tivoli Access Manager, to OpenID or other federated identity systems can be leveraged to protect the data. We can audit who had access to what and encrypt the transport with one- or two-way SSL to prevent eavesdropping. Addressability does not equal vulnerability. In fact, passing references around instead of data is a safer, more scalable strategy. The resource-oriented style is not less secure because it has fewer complicating security features (e.g., XML Encryption, XML Signature, XKMS, XACML, WS-Security, WS-Trust, XrML, etc.), but is arguably more secure because people can actually understand the threat model and how the protection strategies are applied.

These ideas become phenomenally important when we are faced with the daunting and very serious realities of demonstrating regulatory compliance. Credit card companies, health care watchdog organizations, corporate governance auditors, and the like can bring real teeth to a corporate audit demanding proof that only employees whose job function requires access to sensitive information can get to it. Even if your organization is in compliance, if it is difficult to demonstrate this ("First, look in this log on this system and then trace the message flowing through these intermediaries, where it is picked up and processed into a query, as you can see in this other log..."), it can be an expensive process. Employing declarative access control policies against the resolution of logical references makes it explicit (and simple to follow) who knew what, and when.

Data-Driven Applications

Once an organization has gone to the trouble of making its data addressable, there are additional benefits beyond enabling the backend systems to cache results and migrate to new