

Repository Thoughts
S Thorne
February 2005

Here are a few observations, assumptions and conclusions surrounding Repositories.

Users often have an application built to work with a particular repository. They like the application, and wish it would work with other sources of content. People want to use their choice of tool regardless of what repository the content is in, but this is a hard form of interoperability to achieve.

Copying content from one repository to another, or from a repository to an LMS, is not the best solution to providing users content in their preferred environment. Copying content between environments increases the access to this content, but also increases the cost of managing it. Storage, indexing, and updating metadata all add to this cost. Since both the amount and complexity of content is increasing it is important to manage it efficiently. More content needs to be accessed from and managed in a single place to minimize this maintenance cost. This means federation or distribution of these functions as well.

Repositories are diverse and they will continue to be based on different technologies, with different forms of content and management. They can range from massive institutional repositories to a set of files on a users machine. Federated search will be a challenge in this environment, but federating or distributing other tasks, such as managing metadata will be even more important. Applications will need to expect a common form of accessing repository function. Data specifications allow for agreement around content structure and meaning. Protocol specifications define how things work across the network. But if repositories can be local, having a standard network protocol doesn't help an application access them.

Service contracts in the form of APIs may be best suited to define how applications use services. Defining an interoperability point as the service contract allows for more variations in the service implementation. Both parties don't have to agree on what happens on the wire and share a common security model. This is because the service provider supplies the client side service driver that takes care of these details. Of course there still is an actual wire protocol and security, but service providers are free to evolve these independently.

People will continue to want choice, so the specification's landscape needs to expect and provide for this. By defining separately what the data means, from how things work on the network, and from what service functionality an application needs, users can ultimately pick particular combinations of these that suit their need.