Usage Scenarios for Persistent Identifiers

User finds HTML link to PID

User clicks on link and should get back something that makes sense in a browser - either the thing identified or some document about the thing or about access to the thing.

User finds PID in non-linked form

Perhaps less and less common but in various areas of scholarship printed material is still very important. User should be able to recognize that identifier is resolvable, be able to copy/type into browser and then get something useful. Brings in notions of recognizability and brevity/copyability (e.g. long string of digits hard to copy accurately).

Web indexing

We will want resources identified and linked with PIDS to be indexed in popular search engines (Google, Yahoo!, msn, etc.). The URL that the resource is indexed under should be the PID and not the final server (true?). What requirements for redirection etc. should be fulfilled? I think this means that 302 redirect would be appropriate.

Play nicely with services such as VIVO that do content-negotiation to provide data and human views of resources

A client makes a request for a resource identified with a PID indicating that it wants some particular formats (say data in RDF/XML) in the Accept* headers. The client should still get redirected to the appropriate resource which can then support the content-negotiation.

There should probably be some other mechanism (via some other request interface) to get metadata from the resolver (ala handle)

Provide centralized support for Semantic Web / Data views for resources on systems that do not provide for content-negotiation internally

A client makes a request for a resource identified with a PID indicating that it wants some particular formats (say data in RDF/XML) in the Accept* headers. The resolver should understand that it should provide content-negotiation for these resources to either redirect to different URLs depending on the requested type, or to provide an internally generated data view.client should still get redirected to the appropriate resource which can then support the content-negotiation.

Retrofitting a CUL web site with PID's

A website manager is asked to change all the links on a site to PID's. He or she can follow a defined workflow for assigning PID's to links, for modifying location references, and for replacing referenced URL's with PID's.

We need a solution for legacy websites incapable of being modified to store the PIDs created; the resolver must be able to assume that the original URL provided for the resource will continue to function, or that some alternative will be provided when the legacy website ceases to respond. This becomes a matter of policy rather than technology.

Migration of current purl system to new resolver

All existing purls must remain functional (number ? 10-20k??, 1800 indexed in Google). No more would be assigned.

Requires server answering purl dns name and forwarding/responding based on information from new resolver. Would require change of existing workflows to stop assigning new purls.

Support migration of data from staging repository (e.g. Datastar) to long-term storage/preservation repository

The base requirement here is almost the most obvious use case for a PID system: Imagine some data is added to the Datastar repository and is at some later stage migrated to another repository for long-term preservation, it would be good if the id originally assigned were a PID that remained persitent during the migration.

Two possible issues: 1) not all things in the staging repository will later be migrated, some will be deleted. What is the cost of issuing PIDs for items that then get deleted (how many?). 2) migration will likely by a single objects or small groups rather than a whole repository. This must be automated as part of the repository transfer process.

Support for flexibility in access/cache system

It seems likely that access to content in some repository systems will be provided via a caching/access layer on top of the base repository. This is fine in that we can imagine the PID resolver knowing that all access should be directed through that layer. However, for large or frequently requested items there may be a need/desire to serve these from other sources (say cloud storage or even some edge network) to save bandwidth changes and/or improve service. Ideally, use of such facilities would be automatically and dynamically determined (access layer could monitor logs to identify candidates for such treatment).

Redirect to another access/cache service could be done either at the access layer (extra redirect, no implication for PID system) or by some dynamic rewriting of PID resolver data (save redirect, issues of maintenance, coherency etc.). Seems that at least initially one would opt. for the former though the latter is interesting.