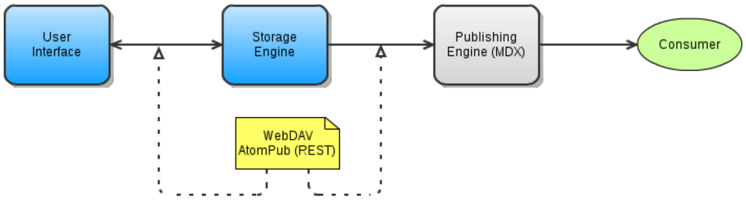
PEER Architecture

PEER consists of 3 parts:

- User Interface
- Storage Engine
- Publishing Engine (MDX)

Each component interacts with the other using a set of protocols (as opposed to a set of APIs) and is interchangeable for alternative implementations of those protocols. A secondary goal of PEER should be to produce a set of components that can be used independently of PEER.



Of these components, 2 are part of the PEER project: UI and Storage, while the third (MDX) is expected to be generated by other efforts. The Storage Engine communicates with the UI using WebDAV and/or AtomPub. If the underlying datastore (i.e the filesystem or VCS inside the Storage Engine) uses a Version Control System (VCS) then each instance of a Storage Engine exposes a specific "branch" of the VCS and each update should generate a "commit".

The Storage Engine is also responsible for wrapping a validation component for metadata - each update is validated before committed to the datastore. The validator component should be pluggable and allow a range of policies for validating metadata to be expressed through code and/or configuration. This ensures that the Storage Engine will be useful outside the context of PEER.

The User Interface should be built with focus on usability and rapid development. It is not meant to be a general UI for everything metadata but should be focused on the PEER user stories. By introducing the Storage Engine and its access protocol(s) it becomes relatively easy to build future alternative frontends, both web-based and others.