

Plugins Promote Preservation (at Penn)

**Using LOCKSS to preserve what you want now,
before the ideal archive does it for you, or it
disappears (whichever comes first)**

**John Mark Ockerbloom
DLF Spring Forum, San Diego
April 14, 2005**

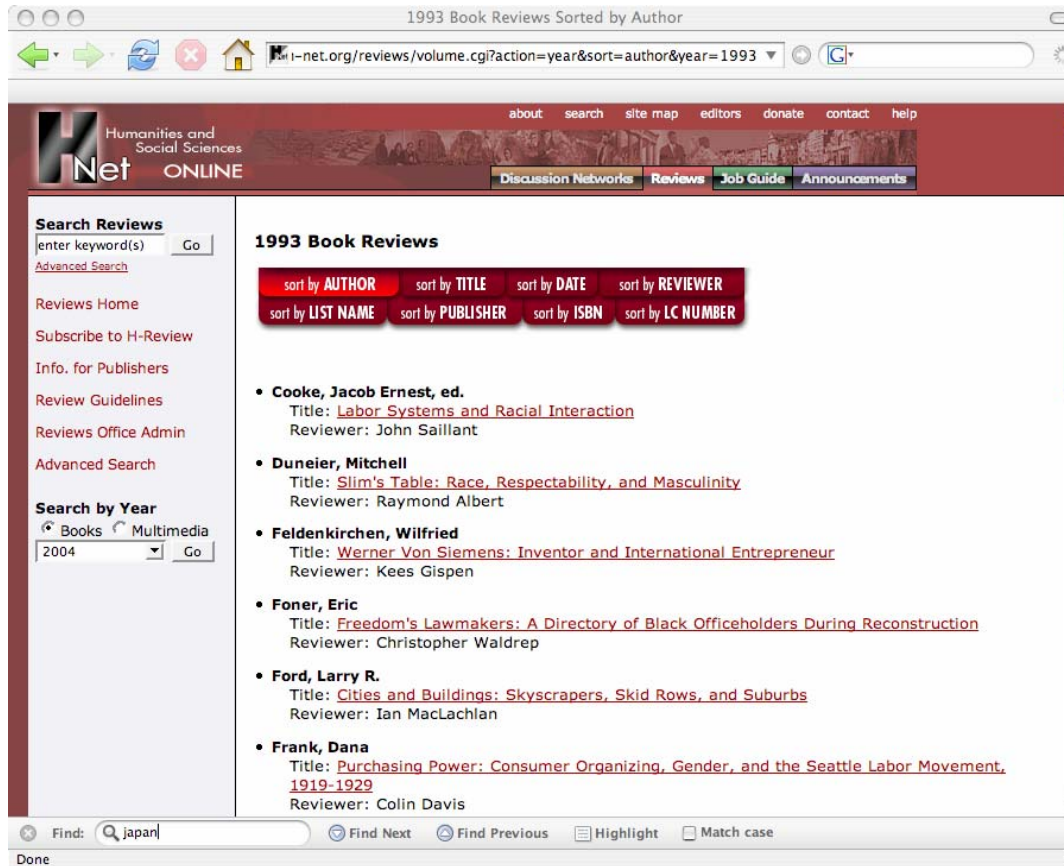
Main ideas

- **The next harvester plugin or manifest not hard to write**
 - The first one can be hard, though: pooled resources may help
 - Overhead for content selection, maintaining harvests nontrivial
 - **New kinds of plugins could also keep content accessible in new formats**
 - TOM now provides tools LOCKSS can use to convert on demand
 - LOCKSS now planning implementation of conversion “plugin” architecture based on TOM
 - **Plugins can also harvest non-journal collections, now**
 - Not difficult to adapt many existing digital document collections for LOCKSS harvesting
 - Would be easier with LOCKSS-ready repositories, better content selection mechanisms
-
- **So what are we waiting for?**

How do we harvest new journals?

- **Publisher provides manifest pages**
 - Pointers to volume contents, plus permission note
- **Someone in LOCKSS provides a plugin to crawl them**
 - If you use the plugin tool, it's just a matter of determining patterns for manifest page, URLs to crawl
 - If you can formulate complex searches (w/ booleans, wildcards, stemming) you can formulate pattern matches
- **Each interested LOCKSS site picks plugins, volumes to acquire**
 - One at a time, or copying a set from someone else
 - Would be nice to automatically acquire “everything with these properties” to allow streamlined local collection control

A LOCKSS-ready journal



Contents pages are manifest pages

Invisible LOCKSS permission note

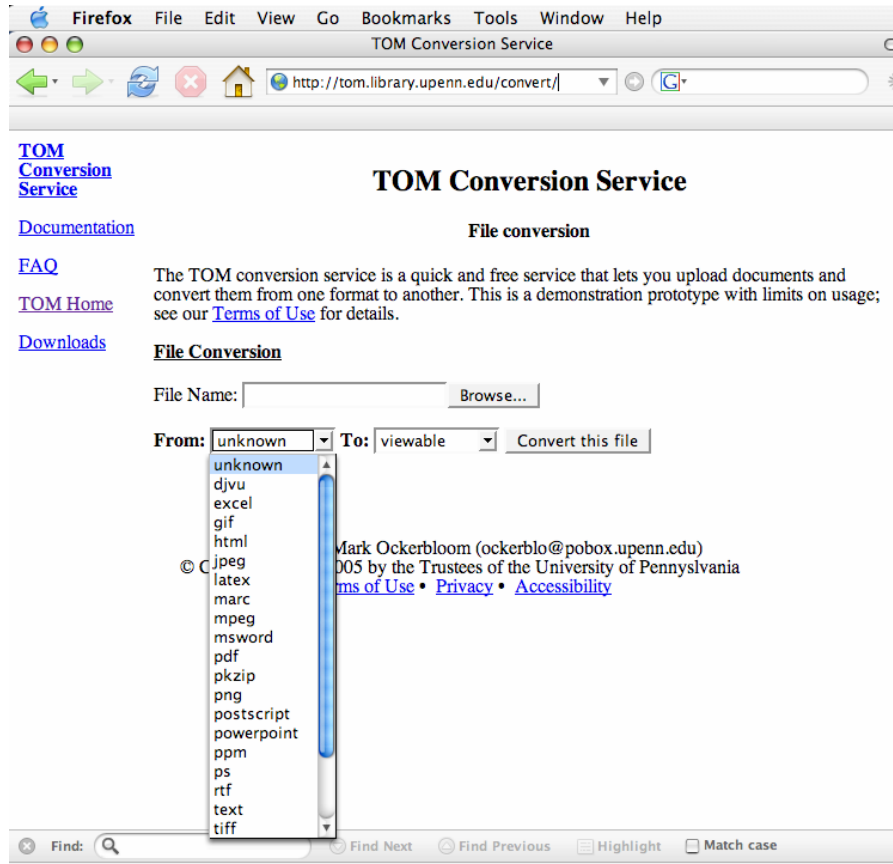
Automatically generated (best way, when possible)

But standalone journals each need new plugin (2 in this case)

How can we keep our harvested content usable?

- **Formats go out of scope over time**
- **LOCKSS plan: keep originals, migrate on the fly according to demand**
- **As formats evolve, will require more and more conversions, intelligent conversion strategies**
- **Fortunately, we've developed systems that can cope with that...**

TOM: open source distributed format support system

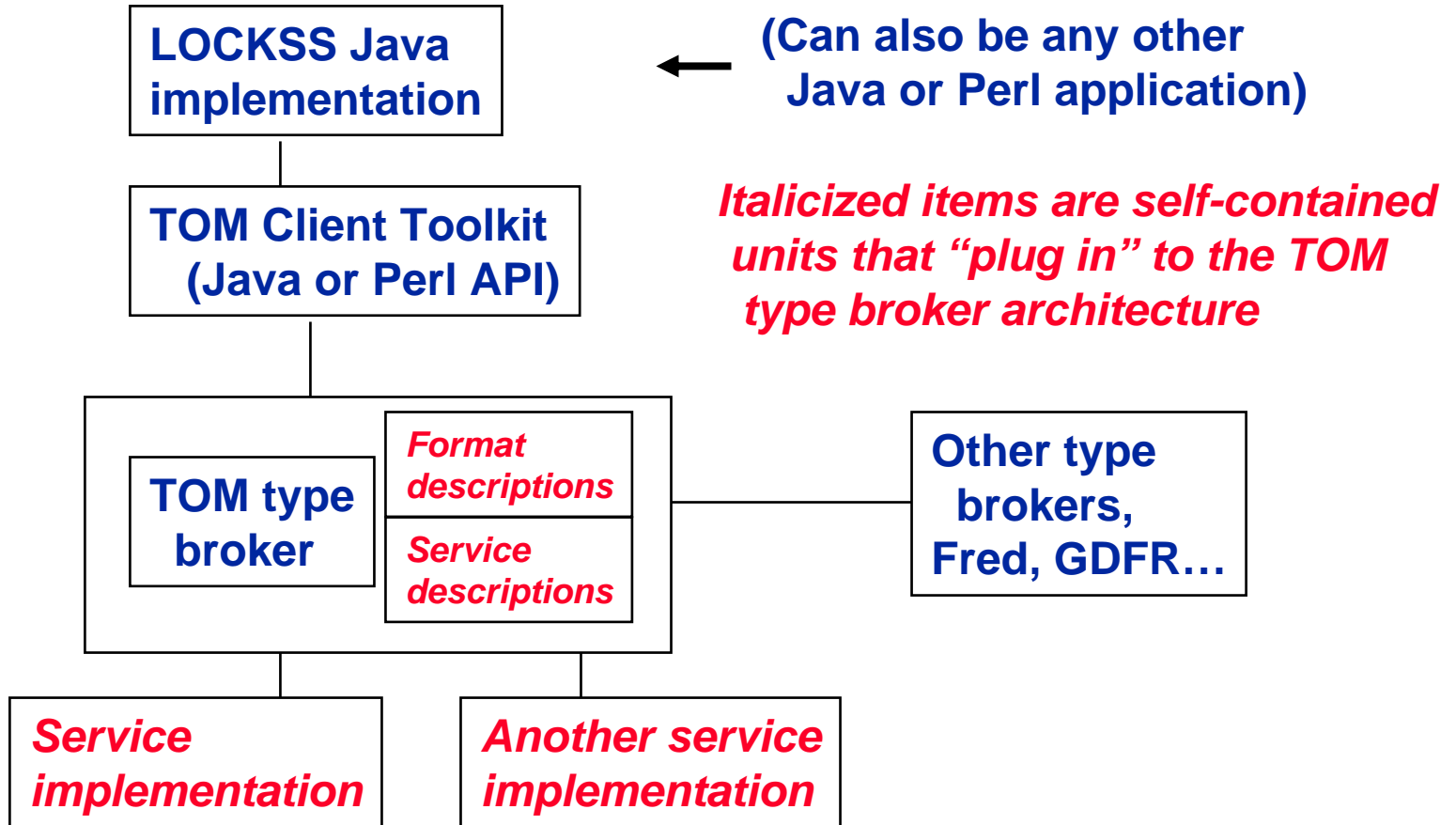


Conversion
Documentation
Program linking
Sharing

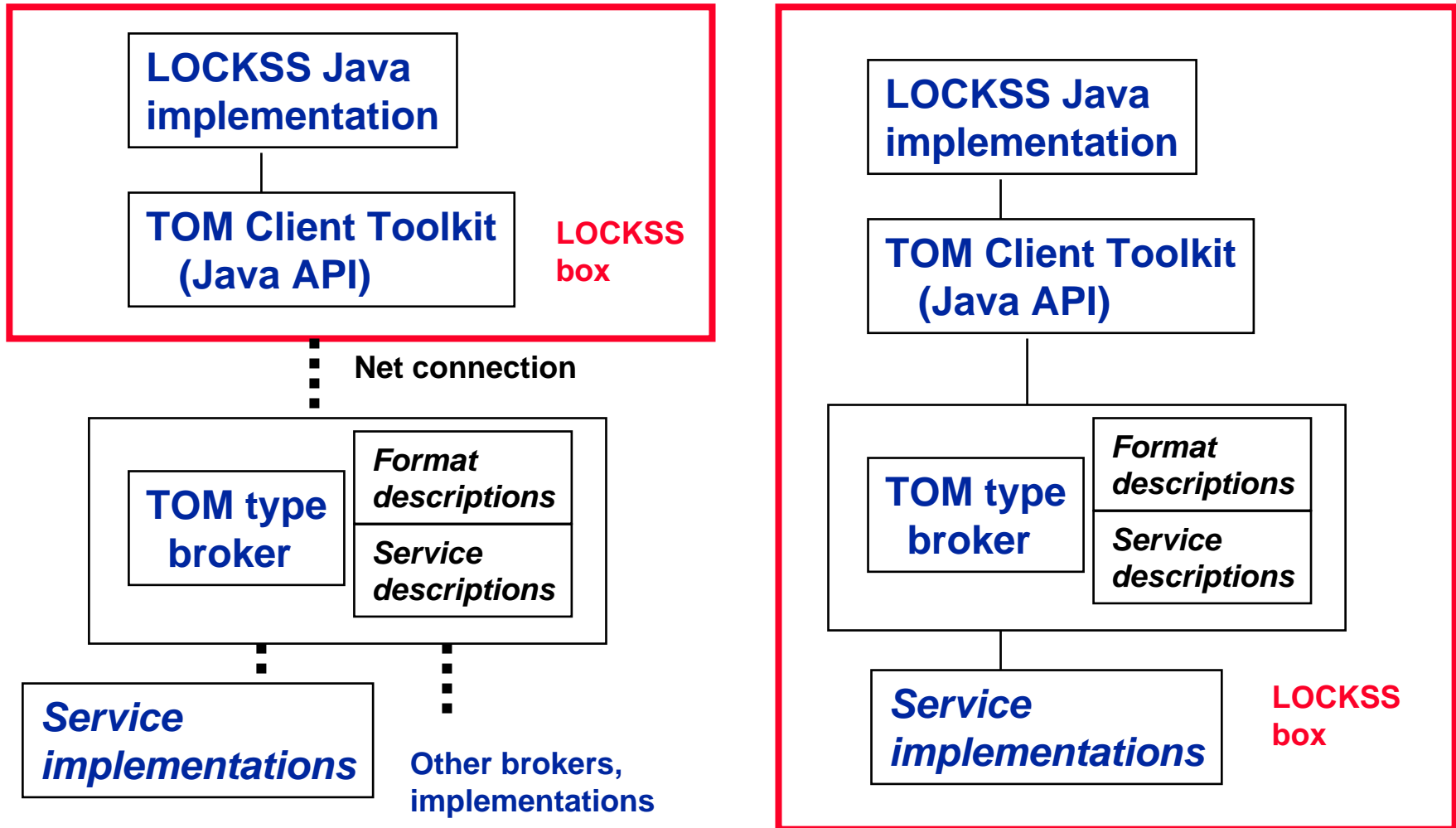
See
tom.library.upenn.edu

Thanks to Mellon
Foundation for support

LOCKSS conversions via TOM



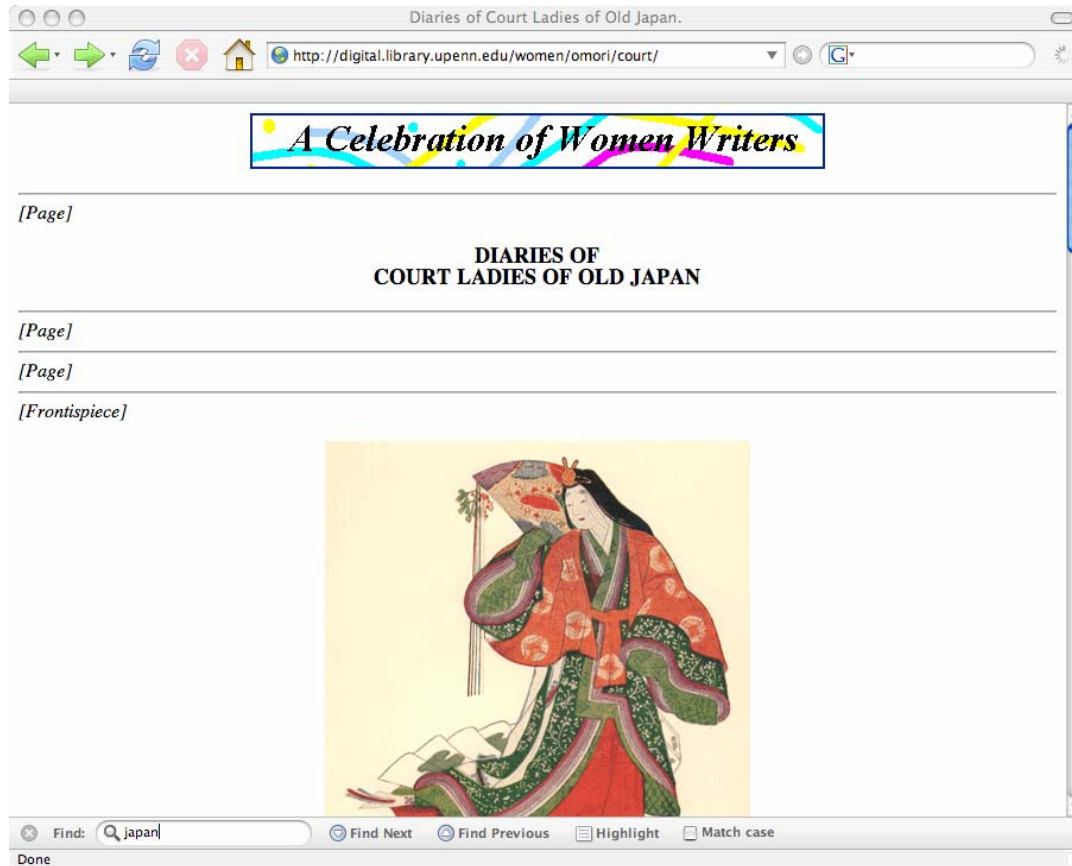
Networked or self-contained



Are we sure we can preserve our own collections?

- **Digital library dark secret: old projects often go offline, temporarily or permanently**
 - A UK project's old journals fell offline after grant ended
 - Many of an Ohio university's digitized rare books still offline after site reorganization months ago
 - These are often *not* picked up by archives in time
 - Even if they go back online eventually, may involve lots of work, long periods of unavailability
- **But the collections often regular enough that one can write LOCKSS plugins for them**
- **Then interested institutions can cache them locally, provide backup access if needed**

LOCKSS-ready books



1 simple plugin for
240-title collection

Existing front pages
(with normalized
URL based on
OAI ID) are the
manifest pages

Invisible LOCKSS
permission note

<1minute/title
to make
LOCKSS-ready
(without automation)

To sum up

- **Once over initial hump, not hard for technical folks to add LOCKSS manifests & plugins for digital content**
 - Reducing number of technical experts needed will help
- **TOM's format "plugins" can help LOCKSS keep content usable over long term**
- **Making collections and repositories LOCKSS-ready can be easy and cheap**
 - Many repository packages now come OAI-ready out of the box; why not make them LOCKSS-ready out of the box too?
 - Other interested LOCKSS caches can then become instant preservation partners by picking up the content
- **Further LOCKSS development needed for adding conversion plugins, scalable collection management**
 - Will happen sooner with support
- **Now is the time to take charge of preserving our stuff**