

# Opening the ILS for Discovery

Interim Report from the DLF's ILS-DI Task Force

**John Mark Ockerbloom**  
**Digital Library Federation Forum**  
**November 6, 2007**



# What's the problem?

- **The “Integrated Library System” is looking more and more to users like Yet Another Silo**
  - Acquisitions, Cataloging, Circulation, OPAC all important to have, and to operate in concert
  - But: need to update what they do, work in context with full range of library and researcher tasks and tools
- **We need Integrating Library Systems**
  - Managing a core set of essential data and services
  - Communicating with other applications to make the most of library resources
- **We need practical solutions soon**
  - The ILS may undergo radical redesign
  - But our users won't wait for that, and we can't

# Integration: Lots of possibilities

- **Some big areas we're not addressing**
  - Acquisitions integration (e.g. w/ financial systems)
  - Cataloging integration (e.g. w/ external cataloging partners inside and outside “librarian” community, multiple forms of catalog data beyond MARC)
  - Item management (physical or digital)
- **Our focus: Patron discovery, from search to use**
  - Finding relevant resources (discovery)
  - Acquiring them (delivery)
  - Managing their usage (patron info and account)

# How did we get here? Where are we going?

- **Spring 2007:** Open discussion session, DLF Forum
- **Summer 2007:** Group formed, work plan laid out, survey drawn up
- **Early fall 2007:** Survey conducted, group met face to face, recommendation outlined, rough draft written, snapshot released
- **You are here:** Rough draft recommendation presented, discussed with Forum community; we gain clarity in priorities, needs, specific technologies, partners
- **Early 2008:** Formal recommendation to be released
- **Beyond?** Recommendation is updated as new technologies, functions, tools emerge

# The ILS-DI group

- John Mark Ockerbloom, Penn (chair)
- David Bucknum, Library of Congress
- Todd Grappone, USC
- Dave Kennedy, University of Maryland
- Emily Lynema, NC State
- Patricia Martin, California Digital Library
- Dianne McCutcheon, National Library of Medicine
- Terry Reese, Oregon State

# Survey

- **Questions, comments about actual and desired use of the ILS, discovery applications that drew on ILS data and services**
  - Responses solicited on DLF-Announce, Code4lib, Ngc4lib
  - Over 150 responses in one week
- **Current use**
  - Majority considering replacing ILS in next 2 years; 1/3 considering FOSS, some considering alternatives (e.g. WC local)
  - Widespread frustration with OPAC interfaces, metadata schemes, resource scope
  - Generally okay with ILS' inventory management functions
- **Beyond the OPAC**
  - 3/4 using supplementary discovery applications
    - » Many locally developed
  - Wide variety of interactions with OPAC
    - » Data export most common
- **More detailed survey result summaries on project Wiki**

# What we're aiming to do: Recommendation scope

- 1. Improve discovery and use of library resources by supporting an open-ended variety of applications that use the data and services of the ILS**
  - We don't specify the applications, just the interfaces they can use
  - Apps may be local or remote, may use more than just one ILS
- 2. Articulate a clear set of interaction expectations for ILS and application developers**
  - Detailed enough to allow clients to “ignore” implementations, implementers to “ignore” client usage
  - Include requirements, inputs, outputs, exceptions...
- 3. Make recommendations applicable to a wide variety of systems and technologies**
  - Avoid locking in transient fads, One-True-Way paradigm
  - Work at 2 levels: abstract **functions/behaviors**  
concrete **bindings**

# Functions

**Abstract but specific description of service or behavior, not tied to any particular technology**

- **Example:**
  - “Return all bibliographic records, with their ids, added to or changed in the ILS since a specified date, in a specified format”
- **Specified: Inputs, outputs, side effects, guarantees, exceptional cases**
- **We also cover some general non-function behaviors**
  - E.g. “Output structured bibliographic data in a configurable pipeline for transforming to display form”
- **Functional areas of interest:**
  - Data aggregation, real-time search/query, delivery, patron information and services, OPAC embed / escape / entry

# Bindings

## Specific technologies that implement desired functions

- **Examples:**
  - **OAI-PMH profile for exporting bibliographic records using marc21 XML and modified internal bibids**
  - **Cocoon server allowing XSLT to be applied to XML schema of bibliographic data**
- **There can be multiple bindings for any given function**
  - **Language-specific object APIs**
  - **Protocols**
  - **Data standards**
  - **Application handoff conventions**
- **Technology examples:**
  - **OAI-PMH, SRW, NCIP, OpenURL, METS, Java/Perl libraries**
  - **Not enough to name technologies; need to specify how they're used to support the desired function (profiles, etc.)**

# What we're aiming to do: Recommendation policy

## 4. Make recommendations that are feasible to implement in reasonable time and cost

- Keep as simple and modular as possible
- With existing ILS where possible, or new systems
- At least one existing-technology binding for each function

## 5. Work with applications beyond “traditional library” domain

- Researchers use applications Not Invented Here to find, organize, work with information
- Interacting with them amplifies library's reach, impact
- Use, but don't require, library-specific technologies (MARC, Z39.50...)

## 6. Be responsive to the user and developer community

- ~~Shamelessly steal~~ Reuse as much work as we can
- Transparency, iterations, open standards&source all help

# Data aggregation functions

- **GetBibliographicRecords**
  - And bibliographic variants:
    - » **GetBibandHoldingsRecords**
    - » **GetHoldingsRecords**
    - » **GetExpandedRecords**
    - » **GetRecord?** (but see also real-time query function)
- **GetAuthorityRecords**
- **Selective export options (by date or by record type) supported**
- **Some possible bindings:**
  - OAI-PMH, object library

# Real-time search/query functions

- **GetAvailability**
- **Search**
- **Scan**
- **SearchAuthorityRecords**
- **ListCourseReserves**
- **ID-based record retrieval:**
  - **GetAuthorityRecords**
  - **GetRecords**
- **Some possible bindings:**
  - **SRU/W, OpenSearch, web service, Z39.50, object library**

# Delivery functions

- **Holds:**
  - HoldTitle
  - HoldItem
- **RecallItem**
- **[Other possibilities: General delivery of copy, in electronic or other format?]**
- **Security, policy issues can be complex for these functions**
- **Some possible bindings:**
  - NCIP, Aleph X-Server, web service, application handoff

# Patron information and service functions

- **Patron identification and credentialing:**
  - LookupPatron
  - AuthenticatePatron
- **Patron information:**
  - GetPatronInfo
  - GetPatronStatus
- **Patron services:**
  - RenewLoan
  - CancelHold
  - CancelRecall
- **Security, policy issues can be complex for these functions**
- **Some possible bindings:**
  - NCIP, Aleph X-Server, application handoff

# OPAC embed / escape / entry

- **OutputRewritablePage** behavior
- **OutputIntermediateFormat** behavior
- **We also need to consider standard ways of getting to particular OPAC views**
  - (may be a bunch of small functions)
- **Some possible bindings:**
  - HTML & Javascript/Ajax, XML&Cocoon (or Ajax), OpenURL, REST

# Getting to an “official” recommendation

- **Make sure we’re not missing any essential functions**
  - And weed the non-essentials (in functions and parameters)
- **Get more specific on functions**
  - (we don’t need to specify all parameters used in practice, just the most important ones)
- **Also find or point to specific bindings of the functions**
  - (reuse and cite what we can)
- **Try to finish in reasonable time**
  - Timeliness may well be more important than completeness at this point
- **Now is the time to encourage implementor involvement**
  - Libraries building on top of existing ILSs
  - ILS vendors implementing interfaces, okaying release of specs and implementations from libraries and third parties
  - Innovators developing rethought ILS services, other collections that could support similar functions

# How can my solution be incorporated?

- **Tell us about it!**
  - Better yet, Point us to public documentation
- **Show how it works as a binding for an abstract function we've specified**
  - or that we should include
- **It has to be openly and fully specified**
- **It ideally has service and client implementations in production**
- **It also helps to have:**
  - No IP encumbrances (e.g. patent) for general open use
  - Open source implementation (especially of client)

# Beyond the recommendation

- **Technologies and tools will continue to be developed**
- **User expectations will continue to increase**
- **Updated (and replacement) designs for the ILS will arise**
  
- **Argues for periodic updates of recommendations to reflect new needs, tools**
  - **But updating overhead needs to be lightweight, while still supporting peer review and fairness**
  - **Would help spur ILS developers to provide what users want**
  - **Is there a sustainability model for this?**

# More information

- See (and comment in) the ILS-DI wiki:
  - <http://project.library.upenn.edu/confluence/display/ilsapi/Home>
- Join the ILS-DI group in our open discussion session later today
- We'd especially like to know about bindings, implementations in active use with released specs
- Email is welcome too: [ockerblo@pobox.upenn.edu](mailto:ockerblo@pobox.upenn.edu) (and say it's for the ILS-DI group)
- Watch for official release in early 2008

Thank you!