Towards a new infrastructure for the World Wide Web

Systems Software Lab

Godmar Back
WHAT YOU BROUGHT TO SEMINAR AND WHAT IT SAYS ABOUT YOU:

Stuff to take notes: First year. Foolishly thinks he'll ever need notes again.

Reading material: Third year. Just here for show.

Didn't bring anything: ABD/Postdoc. Has nothing better to do.

Laptop: Young Assistant Professor. Working on three proposals at the same time.

Playing with latest Gadget/Gizmo: Full Professor. Loooves new toys.
Systems Software Lab (Dr. Back)

- Advanced Client and Server Execution Environments for Cloud Applications
- Real-time Garbage Collection
- LibX platform for libraries
- HPC: Sparse methods
- Simulation
- Educational OS (Pintos)
- Hardware Virtualization for Manycores (VT-ASOS)
- Software Visualization (HDPV)
- Program Analysis
- Domain-specific Languages
- Automatic Program Enhancement
- Operating Systems
- Runtime Systems
- Software Engineering
- Applications
- Real-time Garbage Collection
- Advanced Client and Server Execution Environments for Cloud Applications
- LibX platform for libraries
- HPC: Sparse methods
- Simulation
- Educational OS (Pintos)
- Hardware Virtualization for Manycores (VT-ASOS)
- Software Visualization (HDPV)
- Program Analysis
- Automatic Program Enhancement
- Operating Systems
- Runtime Systems
- Software Engineering
- Applications
Cool projects I won’t talk about today (1)

• **Sparse Methods**
  – How can we make iterative solvers for $A x = b$ faster on current machines?

• collaboration w/ Belgin/Ribbens [ICS07, ICS09, IJPC09]
Cool projects I won’t talk about today (2)

- **Automatic Program Enhancement**
  - How can we let programs complete themselves?
Cool projects I won’t talk about today (3)

• VT-ASOS
  – Can we use hardware virtualization to better support manycore environments?

VT-ASOS collaboration w/ Peng & Nikolopoulous [STMCS ’07], NSF-CSIR 0720673
Cool projects I won’t talk about today (4)

- **HDPV**
  - What do programs really look like?

**A java.lang.String in memory**

**A binary tree being traversed**

collaboration w/ Sundaramanan [ACM SoftVis 08]
Cool projects I won’t talk about today (5)

• The Pintos Educational OS

  – How do we teach OS in a realistic and state-of-the-art manner?

  collaboration w/ Pfaff & Romano [SIGCSE 2009]
A Bit Philosophy

- Applications drive systems
- Users run applications, they don’t care about systems
- Successful systems designers understand applications
LibX: Background

• Brick-and-mortar libraries in the Internet age face a problem
  – Students + researchers forgo library resources
  – Risk becoming irrelevant

• A “virtual librarian” that guides users to library resources while they use the Web
  – integrates access to library resources into the users’ “webflow”
  – no matter which page a user visits (⇒ needs client-side presence!)
LibX 1.0 Features

• Toolbar and right-click context menu
• Adaptive and user-configurable context menus
• OpenURL support
• Magic Button (Google Scholar support)
• Web Localization via Embedded Cues
• Autolinking
• Off-campus access via EZProxy or WAM
• Support for CiteULike
• Support for COinS
• Support for xISBN
• Show/Hide Hotkey
LibX 1.0 Features

• Toolbar and right-click context menu
• Adaptive and user-configurable context menus
• OpenURL support
• Magic Button (Google Scholar support)
• Web Localization via Embedded Cues
• Autolinking
• Off-campus access via EZProxy or WAM
• Support for CiteULike
• Support for COinS
• Support for xISBN
• Show/Hide Hotkey
LibX Timeline

• 2005
  – Released LibX Virginia Tech as a Firefox extension
  – Offered to share LibX with interested libraries

• 2006
  – Tremendous response from library community
  – Received National Leadership Grant from IMLS to create LibX for IE and Edition Builder

• 2007
  – Received LITA Entrepreneurial Award
The LibX Edition Builder

• A configuration management tool for creating customized versions of LibX
  – Customized version of LibX = LibX edition

• Edition configuration includes descriptions of community-local resources:
  – OPACs, OpenURL, Proxy, Databases, Links, Branding, ...

• Edition Builder is easy to use
  – Makes heavy use of OCLC registries
  – Uses sophisticated auto-detection techniques
  – Usable by librarians, not just programmers

• Anybody can create, share, and manage editions

• Over 550 edition as of now, new ones created at a pace of 20/month
  – Huge human investment
  – 10’s of thousands of end users
EDITION BUILDER DEMO
Demo Backup Slide

LibX Edition Builder

Search or browse public LibX editions.
Search by name or description:
Search

Build a new LibX edition from scratch.
You can also go ahead and build your own
from scratch.

Florida International University (FIU)

Bookmarklet

Bookmarklet Name
Florida International University (FIU)

Bookmarklet URL
Template
http://fiu.catalog.fcla.edu/fj.jsp?N=27&Nty=1&Ntt=

Bookmarklet Search
Options
Keyword, Title, Journal Title,
Author, Subject, ISBN/ISSN,
Call Number

Auto Detection: enter the host name of the
machine that runs your library catalog, or enter a
URL to a page that contains a search form. For
instance, try: addison.vt.edu, worldcat.org,
www.citeulike.org, pubmed.gov, hubmed.org, or
complete URLs such as
Enter URL:
OCLC reports that your OPAC may be located at
fiu.catalog.fcla.edu.
probing.fiu.catalog.fcla.edu (1 resource found)
Found Endeca catalog 'Florida International
University (FIU) - Library Catalog' at
http://fiu.catalog.fcla.edu
Architecture

Edition Maintainer

End User

Edition Builder

Database

File System

Web Server

Third Party Resource Servers

OCLC WorldCat Registry

<table>
<thead>
<tr>
<th></th>
<th>Login</th>
<th>Edition and Revision Management</th>
<th>Changes to Configuration</th>
<th>Auto-discovery</th>
<th>Download customized LibX</th>
</tr>
</thead>
</table>

Log Data – Adoption of Edition Builder

- 1155 total editions present by May 2008
- As of Oct 2008, 1600 total editions
  - 460 were made public
- New editions are being made public at a rate of ~20/month

Number of Total Editions since Dec 07, 2007

Days from Dec 07, 2007 to May 7, 2008
Overall Perceived Ease of Use

Overall, you would describe the LibX Edition Builder as:

- Very easy to use: 10%
- Easy to use: 50%
- Somewhat easy to use: 20%
- Somewhat difficult to use: 10%
- Difficult to use: 0%
- Very difficult to use: 0%
In your opinion, the LibX Edition Builder interface is:

- Very easy to learn: 0%
- Easy to learn: 40%
- Somewhat easy to learn: 30%
- Somewhat difficult to learn: 10%
- Difficult to learn: 5%
- Very difficult to learn: 0%
Do you prefer this style of web application to the more traditional, page-based applications?

- I much prefer the LibX Edition Builder style: 40%
- I somewhat prefer the LibX Edition Builder style: 30%
- I do not think the style matters: 20%
- I somewhat prefer the traditional style: 10%
- I much prefer the traditional style: 0%
- No response: 0%
The LibX Edition Builder uses an interaction mode in which configuration changes are immediately saved, so you do not need to press "Save" or "Submit".

In your opinion, this mode of interaction was:
Log Data Results (cont’d)

• 50% editions built in 72 minutes or less
• 80% editions built in 190 minutes or less
Study Findings

• The LibX Edition Builder
  – is easy to use and learn
  – auto-detection is effective at configuring resources

• Created a community

• Open source spirit
  – Anybody can create, share, publish, copy and adapt editions
Where to go from here?

• A toolbar is great, but...

• Emerging technology trends
  – Service-oriented architectures, web services interfaces – soon even to ILS!
  – Data mash-ups; HTML widgets

• Educational trends: librarians, educators, and users create
  – Online tutorials, subject guides, visualizations
  – Social OPACs: tagging, reviews, recommender services
Library Resources and Web Services

LibX 2.0 plugin: executes Libapps, merging library information into pages.

LibX 2.0

Users: decide to which library services to subscribe, see expanded view of the web

Librarians: create or adapt Libapps from reusable, shareable modules

But who will create those modules?
The LibApp Model

• How can the interaction of LibX with web content be modeled?
• Typical tasks involve
  – Examination of the page and extraction of information
  – Processing of information
  – Modification of the page
• A **Module** is a small piece of JavaScript code along with a metadata description of its input and/or output
• A **Libapp** is a group of modules
• A **Package** is a folder of libapps and packages
LibX Tuple Spaces

tuple = TAKE(template)

• If a tuple matching template exists in the tuple space, remove it and return it
• If no tuple exists, wait until a matching tuple is written, remove it and return it

WRITE (tuple)

• Write a tuple into the space
• If a TAKE is pending with a matching template, complete the TAKE

Tuples and Templates are JavaScript objects in JSON notation.
Tuple example: `{ isbn: "0743226720" }`
Template example: `{ isbn : * } // any tuple with ‘isbn’ field`
Tuple Space

{ name: John Grisham }

{ display:
  "published: 1989-2008 (Novelists, American--20th century), most widely held work: The client"
}

Guarded-By: { display: * }
Rationale for Tuple Spaces

Software Engineering
- Low coupling between modules
- Independent composition
- Simplicity
- Suitable for meta-programming

Handling Asynchrony
- User actions happen asynchronously
- Information arrives asynchronously from external sources
- Execution order independence
Back To Systems...
Client Side – Systems Perspective

- Browser’s execution environment must support a mix of components
  - Coming from the original page
  - Coming from widgets included in original page
  - Coming from extensions such as LibX
Why Existing Browsers Don’t Work

• Lack of namespace separation
• Lack of fine-grained security contexts
• Lack of resource management
• Lack of parallelism

```html
<script>
for (var i = 0;; i++)
setTimeout( function () {
    var list = { data: “some string” };
    for (;;)
        list = {
            next: list,
            data: list.data + list.data
        }, i++);
}, i++);
</script>
```
Script Spaces

• Script space represents execution context
  – May be Tab, Page, Frame, Extension module or Part of Page

• Unit of resource consumption
  – CPU + Memory + Garbage Collection

• Separate security context

• Separately schedulable
Server Side – Systems Perspective

- Many apps use Web as their primary user interface ("AJAX")
  - E.g. LibX Edition Builder
  - Google Mail, Docs, etc.
- Must manage complex UI state
Challenges for Server-side Environments

• Web designed for static documents
  – Interactivity was retrofitted
• Stateless nature of HTTP
  – Must re-render complete presentation state on every request
• If state is kept on client, developer must manage client-server communication
  – Writing distributed applications is hard
Idea: A “remote-display” approach

• Server-centric AJAX Framework
  – Keep presentation state on server side
  – Use aspect-oriented techniques to track changes to presentation state
  – Render on demand

• Drastically simplified design
Conclusion

• To build systems, must understand applications
• Not just theoretically, build and support them
Acknowledgements

LibX Team
• Annette Bailey
• Godmar Back
• Kyrille Goldbeck
• Michael Doyle
• Arif Khokar
• Travis Webb
• Alumni
  – Nathan Baker
  – Tilottama Gaat
  – Tobias Wieschnowsky
  – Robert Ellis

• Advanced Execution Environments
  – Amarjyoti Deka
  – Michael Doyle

• Students
  – Mehmet Belgin
  – Peng Lu
  – Ben Pfaff
  – Anthony Romano
  – Jaishankar Sundararaman

• Faculty
  – Dimitris Nikolopoulos
  – Calvin Ribbens
  – Eli Tilevich
I’m hiring for multiple open GRA and URA positions