From the Programmer’s Point of View: Imagining Creative Solutions to Serve our Patrons

ACRL New England 2017 Annual Conference
2017-05-12
David Cirella
Emerging Technology Librarian
New York Institute of Technology
A problem is the first phase of development

Given enough time/resources/support any (computer) problem can be solved
What this talk is not

- Not a call for librarians to stop being librarians
- Not a call for everyone to become a programmer
- Not a statement about the future of libraries
What this talk is meant to be

- A call for librarian empowerment

- A call for all types of librarians to take an active role in the design and implementation of software services
Standing on the Shoulders of Giants
Standing on the Shoulders of Giants

- The inspiring work of systems librarians and library developers from all types of libraries
Custom Software Solutions

• We begin with:
  – Small applications
  – “glue” between larger systems

• In the future:
  – Contributions to larger open source projects
  – Collaborative projects
Different Paths

- People who program
- People who need to tell the programmer what to do
What’s the goal?

- Give the user what they want, how they want it

- User experience trends are set by commercial entities
  - Expectation for library services to have similar user experience as non-library applications
What’s the goal?

- Satisfy the user’s information need in as few steps (clicks) as possible
  - Generally with some software somewhere in this process
ACRL top trends

• Research data services (RDS)
• Data policies and data management plans
• Digital scholarship
• Open Educational Resources (OER) [1]
Ongoing Shift

• ARL Statistical Trends
  - Reference Transactions (-77%) 1991-2015
  - Initial Circulation(-58%) [2]
  - Graduate Students (+149%) 1986-2015
  - Total Students (+54%) Faculty (+40%)
  - Interlibrary Lending (+82%) Interlibrary Borrowing (+237%) [3]
Ongoing Shift

- **ARL Statistical Trends**
  - Ongoing Resource Expenditures (+521%) 1991-2015
  - Expenditures for Bibl. Utilities, Networks, etc. (+411%)
  vs
  - TOTAL Expenditures (+197%)
  - One-Time Resource Expenditures (+79%) [4]
How are we doing it today?

- Online catalogs
- Discovery Layers
- Online research guides
- Mobile Apps
- Repositories, Digital Collections
- Custom library applications / websites
Traditional Options

- Off-the-shelf commercial products
- Ask institutional department (IT, Communications/Marketing)
- Large-scale open source projects
Downsides

- No perfect solution for our problem
- Time
- Cost
- Reliance on others
No perfect solution for *our* problem

- Existing, off-the-shelf software not library oriented
- The specific need may be:
  - Too small
  - Too specialized
  - Too undefined
Time

- Procurement process
- Long build times
- Not responsive to users’ needs within common time frames
  - Semester
  - Academic Year
Cost

• Commercial solutions have recurring costs (maintenance contracts)
• Hourly rates for new development or features
• Funding for software is often in competition against acquiring resources / ongoing subscriptions
Reliance on Others

- Others’ concept of what our users need
- Maintenance
- New features
- Responsiveness
Moving Forward

If core library services are to be delivered exclusively through web applications

And our role is to get people the resources they need

Then we should have a very large role in the design of those applications
Solution

DIY – Do it Yourself – Do it Ourselves

• We know our users’ needs best
• We know what would make things better
• We can do this ourselves
What do we need to Do it Ourselves?

• Knowledge:
  – Software Development Process
  – System Architecture

• Skills:
  – Programming
Software Development Process

- Specification
- Design & Implementation
- Validation & Testing
- Evolution \[^5\]
Case study – Reserves Kiosk

• What we have:
  – Listing of reserve items kept in a spreadsheet, printed and placed in a binder on the circulation desk.

What we want:
  – Touchscreen kiosk
  – Off-campus availability
  – Easy/fast to maintain and update
Software Development Process

• **Specification**
  - Define the problem
  - What features are required to solve the problem?
  - Use cases / User stories
Case study – Reserves Kiosk

• **Specification**
  - Define the problem
    • We need a listing of reserve collections
    • It needs to be easy to use, modern, and not introduce a major new service
  - What features are required to solve the problem?
    • Online availability
    • Provide a kiosk-like interface
    • Simple (cheap) to deploy
    • Short development time, low resources
Case study – Reserves Kiosk

• Specification
  – Use cases / User stories
    • As a student, I need to know if the textbook for my class is on reserve
    • As library staff, I need to be able to quickly update our list of holdings
Software Development Process

● Design & Implementation
  - Identify architecture and application components
  - Evaluate software languages, existing code libraries
  - Target deployment platforms
  - Consider existing data and systems
Software Development Process

Architecture of an Application
- Presentation Layer (User Interface)
- Application/Business Logic Layer
- Data Layer (Database, Information store)
- Platform
Case study – Reserves Kiosk

Architecture of an Application

- **Presentation Layer**
  (User Interface)
  - Web-based, standards compliant
  - Touch, app-like

- **Application/Business Logic Layer**
  - Interface with Google Apps
  - Fast, simple

- **Data Layer**
  (Information store)
  - Relational Database
  - Easy to host

- **Platform**
  - Low upkeep, low cost
Case study – Reserves Kiosk

• Design & Implementation
  - Evaluation of software languages, existing code libraries
    • Open source, leverage existing solutions, code
    • Bootstrap (HTML, CSS, Javascript)
    • Python Flask (Web framework)
    • Postgres Database
  - Target deployment platforms
    • Low cost shared web hosting
    • iPad
  - Consider existing data and systems
    • Leverage current reserve lists
    • Utilize Google Apps (secure, staff already using it)
Case study – Reserves Kiosk

- Information Flow

Front End

- Web Interface
- iPad

Back End

- Flask Web Framework
- Database
- Web Host

External Resources

- Google Sheets
Software Design

• Validation and Testing
  – Simulation Testing
  – Component Testing
  – User tests

• Evolution
  – Ongoing maintenance
  – Changing Requirements
  – New features
  – Change Tolerance
Case study – Reserves Kiosk

• Validation and Testing
  – Simulation Testing
    • Sample data
  – Component Testing
  – User tests
    • Informal discussion with users
    • Participant observation

• Evolution
  – Ongoing maintenance
    • Updates
    • Evolving platforms
  – New Features
    • Additional locations
    • New types of resources
Case study – Reserves Kiosk
Getting Started with Programming

- Work with Web standards
  - HTML
  - CSS
Getting Started with Programming

• Pick a Language, Any Language
  – Python
  – Javascript
  – Ruby
Getting Started with Programming

- You will Eventually need a Database
  - Postgres
  - MySQL
Getting Started with Programming

- Learn a common platform for deployment
  - Unix / Linux
  - Cloud / Web
Getting Started with Programming

Front End

HTML
CSS
Javascript

Back End

Postgres

Python

External Resources

Linux

Google API
Another Project

- **3D Printer Tweetbot**
  - **Goal:** Provide interactive real time updates on status of 3D Prints
  - **Design:** Twitter as the public front end / user interface
    - Python to interact with Twitter API
    - Raspberry Pi as platform
  - User tweets a trigger word to the bot account, gets reply with real time photo of 3D printer bed
Thanks!
References


Image Credits

Prayitno- Crossroad https://flic.kr/p/7G3Klb