6.170 – Software Studio

Ur/Web:
A Simple Model for Programming the Web

Adam Chlipala, Mark Day, Daniel Jackson
November 24, 2014
Ur / Web

**Ur**
A new general-purpose typed functional language

**Web**
Tools for implementing modern three-tier web applications

*Motto:* “the compiler understands the structure of your web app.”

*Focuses on:*
- Programmer productivity
- Security
- Performance (especially server-side, for scaling)
There are at least several users whom I have never met!

The first commercial Ur/Web application: *BazQux Reader*, by Vladimir Shabanov

Feed reader with comments

http://www.bazqux.com/

On the order of 1000 paying users daily
Ur/Web is distinguished by

a high-level **programming model**,

a good **security** story,

and unusually good server-side **performance**.

And programmers who get along well with Haskell will be extremely **productive** with it, too!
Programming model?

Code with explicit static types for all the key pieces of a web app.

Compiler tells you when you've screwed up.

Ur has a very expressive type system underlying its presentation of, e.g., a library for compile-time checking of HTML validity. Explaining the type system would take more time than I have!
Security?

The most common security vulnerabilities are impossible by construction. Say goodbye to code injection attacks and cross-site request forgery.
Performance?

Aggressive, domain-specific optimizing compiler produces native code that is decently likely to be more efficient than hand-coded C that you'd have the patience to write.

*Caveat: the Ur/Web concurrency model is simpler than in most frameworks, and there is a performance cost. More later....
TechEmpower Web Framework Benchmarks

- 3rd-party benchmarking initiative
- 93 frameworks in test to the right
- Ur/Web's performance:
  - 4th best throughput (~100 krps)
  - Best latency (2.1 ms)
- A new optimization should improve results for next round :-)

Fortunes

Results

Best fortunes responses per second, Dell R720xd dual-Xeon E5 v2 + 10 GbE (104 tests)
What web programmers expect today

Web Server → Database

HTTP, HTML, URLs

Web Server → Browser

HTTP, HTML, URLs

browser

Web Server → Browser

HTTP, XML/JSON

Database

SQL

Database → Browser

HTTP, XML/JSON

Browser

JavaScript

The DOM
(a big old mutable global variable, modified by looking up nodes by string ID)

Browser

Ajax!

Comet (long polling) or WebSockets

All represented as normal strings in most frameworks!

Hello, code injection attacks!
The Ur/Web model

An application is written entirely in Ur/Web source code, exposing this view:

- **Web Server**
- **Database**
- **Browser**
- **Thread**

**Key point 1:** threads are spawned in Ur/Web code and themselves run Ur/Web code.

**Key point 2:** client-side threading is **cooperative**, with thread switches only at well-defined points (a natural choice in JavaScript).

**Typed message-passing channels** stored in DB and passed to clients.

- **List of records of native Ur values**
- **Strongly typed document tree**
- **First-class function to run on server**
- **Native Ur value**

**Key point:** semantics say first-class requests are run **one at a time** (the standard transaction abstraction).

**Key point:** semantics say first-class requests are run **one at a time** (the standard transaction abstraction).

**Strongly typed SQL syntax tree**

**Typed message-passing channels** stored in DB and passed to clients.

**Document**

- **Mutation function for rendering based on contents**

**Thread**

**Thread**

**Thread**
Demo time!

Open source at:
http://www.impredicative.com/ur/