Get Rich Now!
Selecting A Rich Client Technology

Vladimir Vivien
Software Engineer

Objective

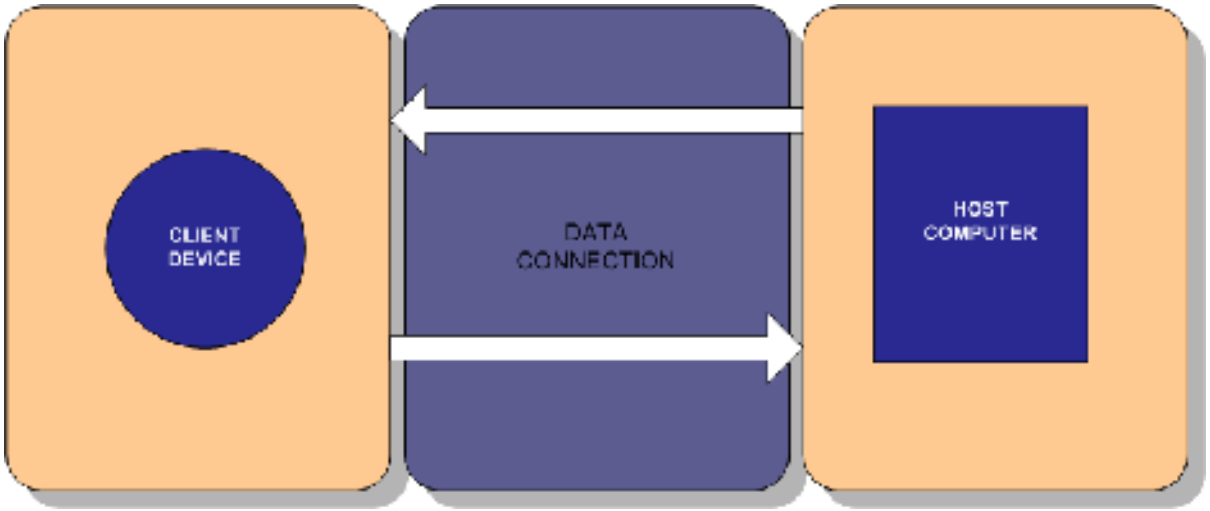
Provide an overview of emerging Rich Client technology options available.

Agenda

- Client / Server Models
- Understand Rich Clients
- AJAX
- XUL
- OpenLaszl and Flex Flash Platforms
- Eclipse and Netbeans Desktop Frameworks

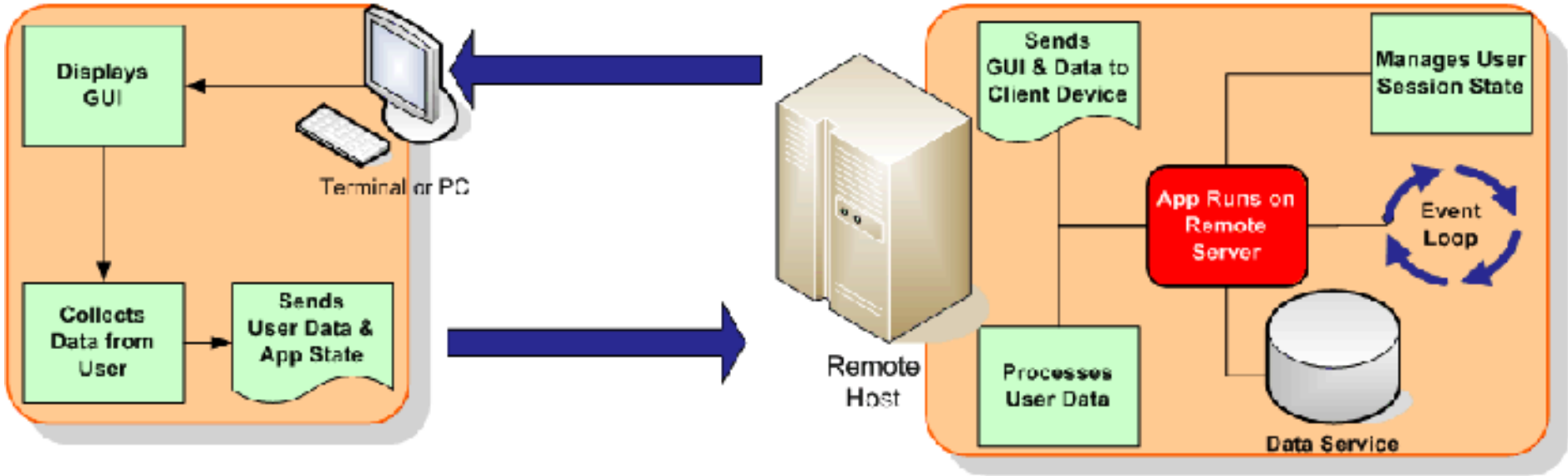
Client / Server Models

Client/Server Model Review



- Client and remote host communicate over an established data link.
- User interacts with the client devices when using applications.
- Remote host provides data and sometimes GUI services to client.

Thin Client Model

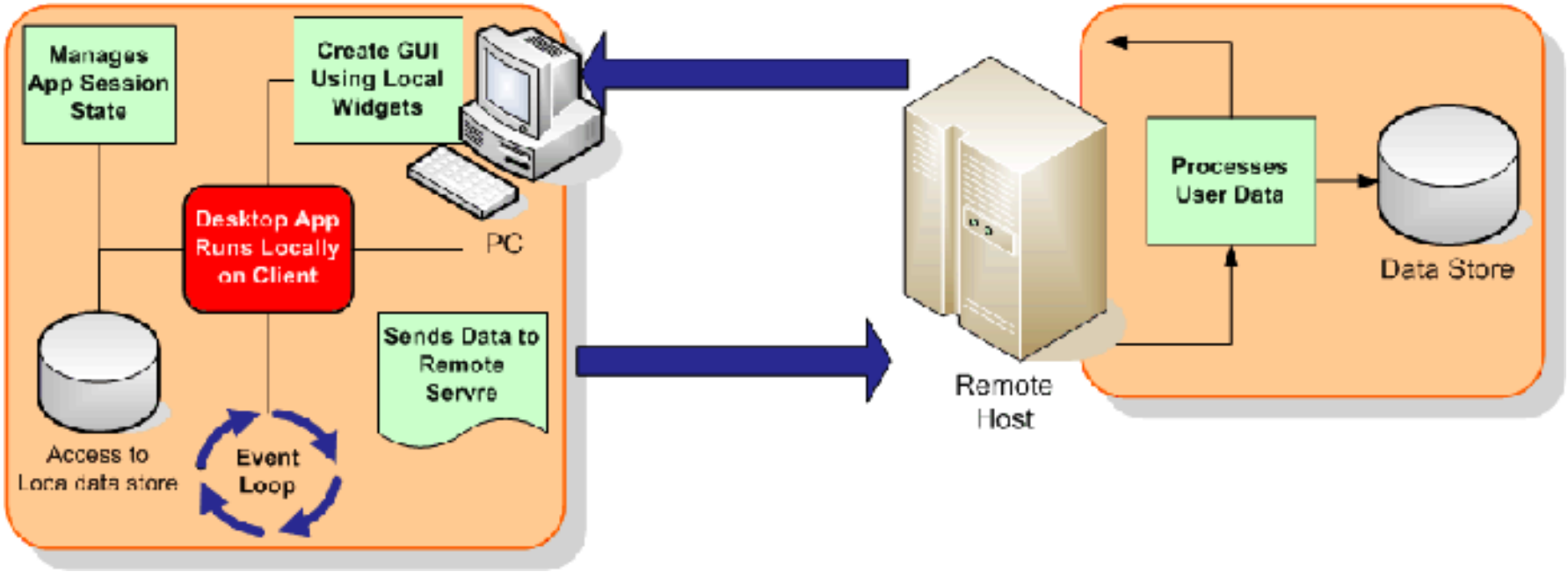


- Remote computer handles
  - Application UI state
    - In dumb client model, provides entire OS
  - Business logic / domain data
  - Data storage / retrieval
  - Event Loop

Thin Client Model (2)

- Specialized software (or hardware) provides communication services to remote host computer.
- Application has no access to local access resources.
  - In dumb client model, no local OS is available.
- UI events are sent to server for processing.
  - Creates latency and performance issues.
- Application Deployment Model
  - Access to application is just-in-time.
  - Application runs on remote computer.
  - Zero-installation of application on client devices.
  - Upgrades occur on server not client devices.

Fat Client Model

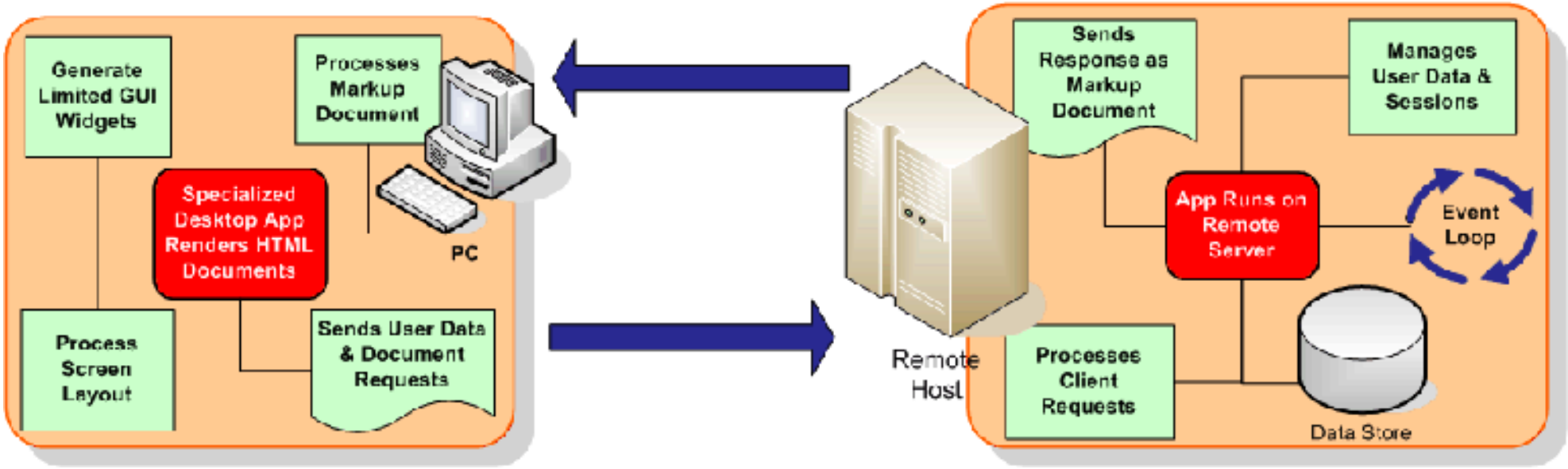


- Desktop application that runs on local PC
  - Manages GUI and event loop
  - Processing of business logic
  - Ability to save/retrieve data locally

Fat Client Model (2)

- Application communicates with backend data server.
  - Ability to function both on- and off-line.
- Access to processor, graphics cards, and periphery devices.
- Uses graphical components including menus, trees, data grids, calendar controls, etc.
  - User-friendly and increase usability
- UI events are handled directly on the device
  - Little latency handling user gestures.
  - Increase in performance
- Application Deployment Model
  - Application must be installed on each client device.
  - Upgrades must occur on each device.

Web Browser Model

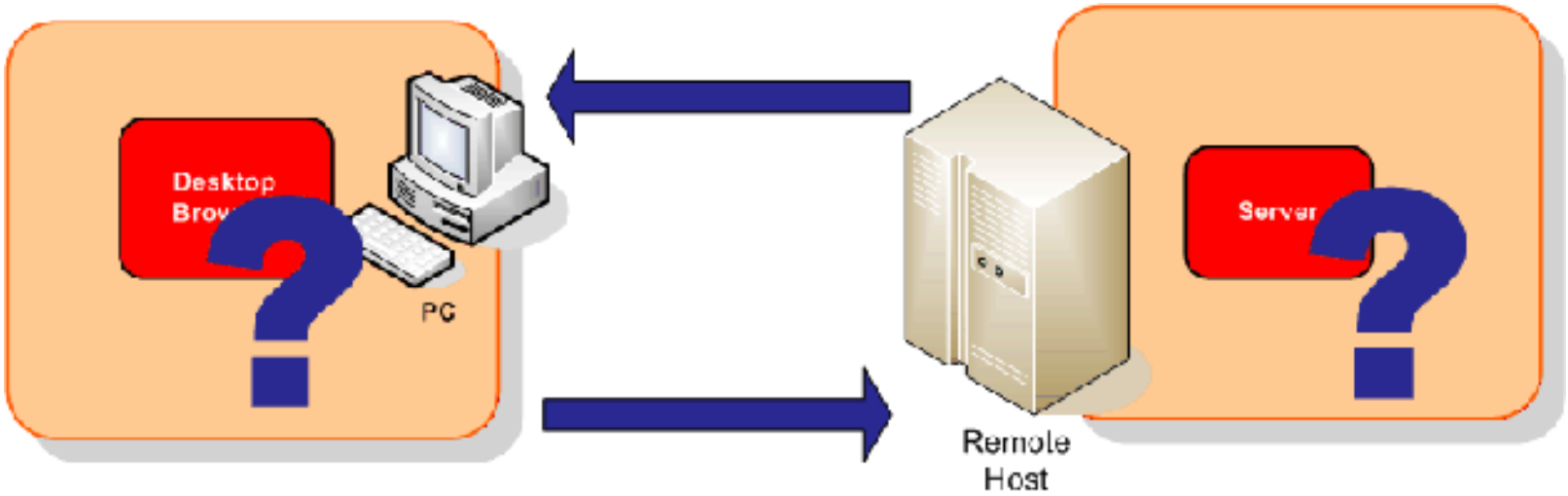


- Specialized desktop application
  - Simple request/response communication model using HTTP.
  - No state persistence between request (stateless).
  - Request fetches document-based content from server.
  - Renders, formats, and lays out content on screen.
  - Generic container for hosting document or application.

Web Browser Model (2)

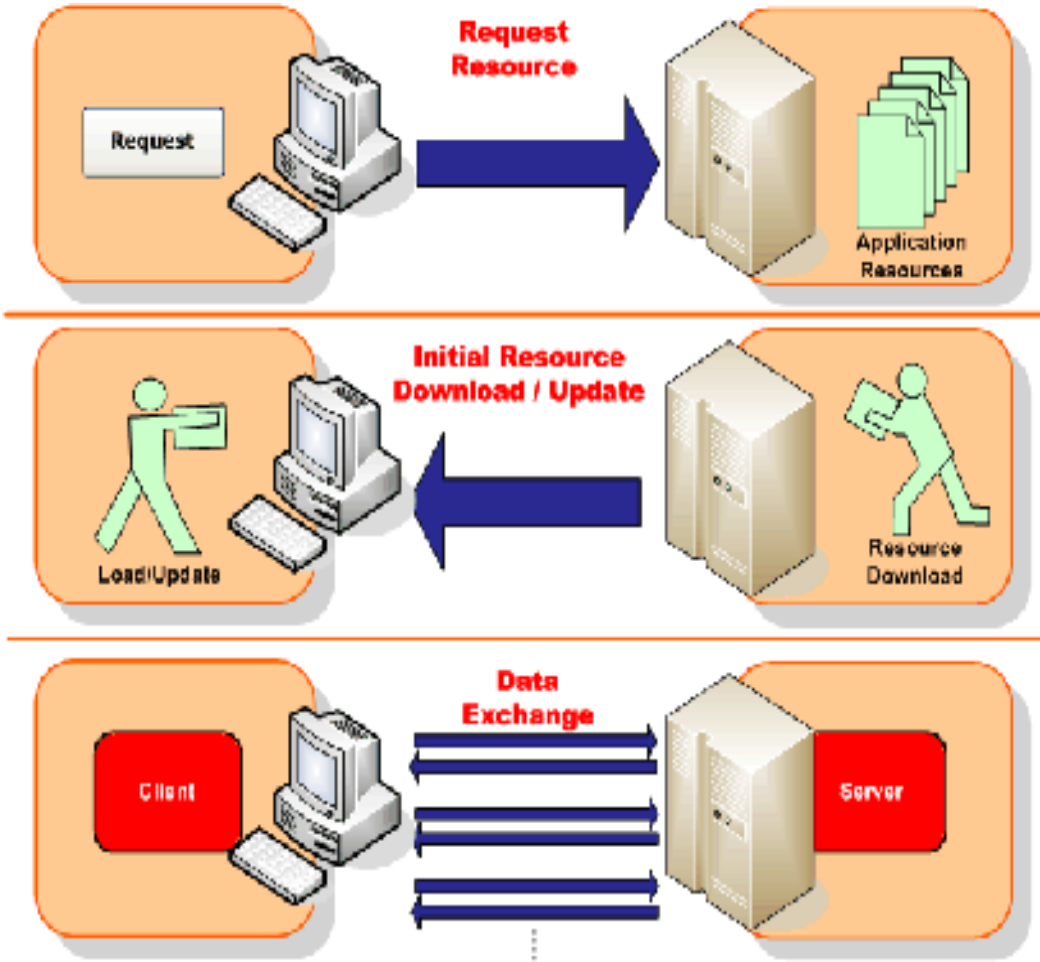
- Application Runs on Server
  - Extension of thin client/server model.
  - State/session management achieved by token sharing.
  - Creates a perceived event-loop on server.
- Coarse event cycle
  - UI events causes full document request cycle to re-render screen.
  - Creates latency and performance issues.
- Access to limited GUI controls
  - Handles textual form-based data entry.
  - Decreased user-friendliness and usability
- Deployment Model
  - Zero-install on client
  - Upgrades occur on server.

What Is A Rich Client?



- *High Usability:* user-friendliness of fat desktop clients.
- *Snappy*: Granular client-side event cycle.
- *Well Connected*: leverages HTTP's for server communication.
- *Device Fidelity*: targets specific device capabilities.
- *Zero-Install:* ability to automatically update.
- *SOA-Compliant*: first-class citizen in service architecture.

Rich Client Phases



- Phase 1 – Request Resource
  - Client requests application resources from server
- Phase 2 – Asset Provisioning
  - One-time resource download/update
  - Include data, code, media
- Phase – Normal Usage
  - Consist only of data exchanges
  - No further GUI download

AJAX

AJAX

- Asynchronous JavaScript And XML
- Formerly Known as DHTML
- Pattern for Web-Based Development
  - Standard web stack: *HTML*, *JavaScript*, *XML*, *CSS*, *DOM*
  - Asynchronous event model through JavaScript
  - Programmatic control of component's style, data, & layout
- Browser-Based Technology
  - Runs on modern browsers: *Firefox*, *Opera*, *IE*, *Safari*
  - Application rendered within web browser
  - Zero-Install / Zero-Update costs on client device
  - Security constraints – no access to local storage

AJAX Frameworks

- Rich UI Widgets
  - Calendars, sliders, trees, data grids, tabs, calendars, menus
  - “Cinematic interactivity” and drag-n-drop
- Server-side support: Java, PHP, Python, and Ruby
- Abstraction Event Models & Browser Capabilities
- Popular Open Source Frameworks
  - *Prototype*, *Scriptaculous*, *Rico*, *Dojo*
  - Zimbra: Rich and re-usable UI components
  - Yahoo API: *Mapping, searching, GUI widgets*
  - Google API:*Mapping, searching, calendering, data storage*
  - DRW: *JavaScript data-remoting API for Java*
  - *And much more ...*

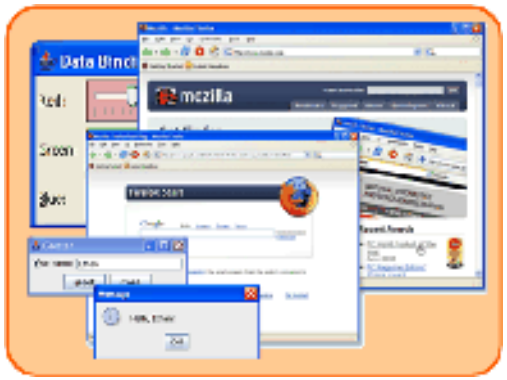
AJAX Code Compilers/Emitters

- Generated JavaScript
  - Use other languages to generate JavaScript
  - Provide JavaScript abstraction constructs for windowing, widgets, animation, data-remoting and event handling.
- Frameworks
  - Echo2: Server-side Java
  - Google Web Toolkit: Compiles Java into JavaScript
  - Java Server Faces: Server-side Java
  - OpenLaszlo Legals (featured): Tag-based XML language

AJAX Considerations

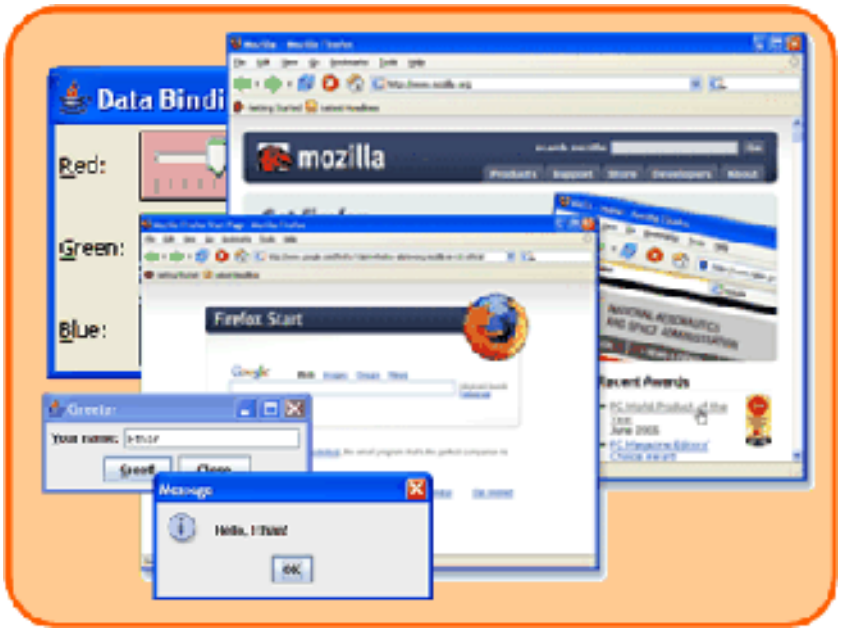
- JavaScript skills are required
- Knowledge of HTML, CSS, and DOM are necessary.
- Start with a simple framework like Rico
- No standard component models / risk of lock-in
- Code can be viewed from browser
- Both open source and commercial options available
- IDE support non-existent for most open source options.
- Rich set of visual web services:
  - Mapping, calendaring, advertisement, video, content, etc
- Excellent platform for composite applications (mashups)

XUL



XUL

- eXtensible User-interface Language
- XML for GUI Description
  - Specifies screen element layout
  - Style through CSS
  - Provides scripting language
  - Facilitates Data Binding
- Compiled XML
  - Compiled into executable for a runtime engine
  - Runs as stand-alone or hosted in a browser
- Diverse Family of Runtime Environments
  - Mozilla Browser, Flash, Java, and MS.Net.



XUL On Mozilla

- Extension of Standard Web Technologies
  - Uses JavaScript, CSS, DOM, and XML
  - *Native look-and-feel* components:
    - *Menus, popup menus, lists, trees, grids, panels and tabs*.
- Mozilla Proven
  - Firefox and Thunderbird are XUL applications
- Support Browser-Based Deployment
  - XUL files downloaded and rendered within browser
  - Zero-Install / Zero-Update costs
- Support Stand-Alone Deployment
  - XULRunner runtime tools
  - Installation and upgrade management

XUL: Other Family Members

- Java
  - JAXX: compiled into Swing
  - SwiXml: compiled / Applet runtime
  - Thinlet: Applet runtime
  - Canoo's ULC XML: Applet runtime
- MS XAML – .Net Windows Forms
- Flash (featured)
  - Flex
  - OpenLaszlo

XUL Considerations

- No standard XUL implementations / risk vendor lockin
- Requires learning new XML language constructs
- Documentation maybe an issue for non-commercial offerings
- Minimal open source IDE support
- Select XUL platform closer to targeted runtime
- Mozilla XUL
  - Leverage knowledge of web technologies
  - Plug-in architecture for feature scaling
  - Skinable look-n-feel
  - Runtime available for several OS's (Windows, Linux, Mac)
  - More features in Firefox = more features for your apps

Flash
Rich Client Platforms

Flash

- Vector Multimedia Runtime
  - Produced by Adobe (obtained from Macromedia)
  - Compressed small vector file format (swf)
  - From time-based movie paradigm to desktop app model
  - Support for multimedia assets and animated components
- Flash-Based Rich Applications
  - Desktop-like GUI with discrete component event cycles
  - *Browser-hosted:* Zero-Install / Zero-Update costs
  - Browser plug in with large install base (97%)
  - Ability for stand-alone deployment (via Flash Player)
  - Abstraction of event handling, animation, and data services
  - Rich visual components: trees, menus, sliders, grids, etc

Flash Rich Client Platforms

OpenLaszlo

- Open source project from Laszlo Systems
- Generates Flash binaries
- New version generates AJAX
- Laszlo Platform
  - Cinematic GUI widgets
  - OO-based declarative XML language: LZX
  - Scripted with ECMAScript (standard JavaScript)
  - *Customizable controls*: data grid, tree, menus, tabs
  - Web-based runtime and admin tools

OpenLaszlo Tools

- Open Laszlo Framework
  - Full SDK (written in Java)
  - Runtime compilation of LZX into
    - Flash binaries (version 7, 8, and 9)
    - AJAX / DHTML files
    - Scalable Vector Graphics (SVG, in early phase)
    - Plans for others
- Laszlo Services
  - Just-in-time compilation via Laszlo Servlet
  - *Data Services:* caching, optimization, data remoting
  - Inherent support of SOAP, XML-RPC, JavaRPC
  - Built-in security service

Laszlo Considerations

- Leverage knowledge of JavaScript
- Entire technology stack is open source / free
- Excellent documentation / tutorials / training available
- Laszlo Legals targets multiple runtime engines
- Provides comparable features as Flex for free
- Default GUI components not crisp / dated look
- Extensible object-oriented API and components
- No Standard CSS model
- Slow, but growing adoption
- No third party component market place
- Not standard / risk of vendor lockin

Introducing Flex

Adobe Flex

- Adobe Flex from Adobe System Software
- Commercial Platform
  - SDK (only tool free)
  - Flex Builder IDE (Eclipse-based visual tool)
  - Flex Charting
  - Flex Data Services
- Flex Platform
  - MXML: XML-based language for GUI layout
  - Scripted with full OO ActionScript (ECMAScript)
  - *Extensible components*: data grid, tree, menus, tabs
  - Implementation of CSS for GUI styling

Flex Tools

Adobe Flex

- Flex Framework
  - Compile-ahead model – then deploy
  - MXML/ActionScript Compilation
    - SWF binaries only
    - ActionScript 3 compiled to native code at runtime
- Data Services
  - RCP / WebService proxies
  - Data synchronization
  - Messaging for ESB integration
  - Just-in-time compilation through URL requests

Flex Considerations

- Easy to get started with SDK
- Extensive documentation/ tutorials / training
- Leverage of Adobe and Macromedia mindshare
- Support for enterprise environment
- MXML is a simpler abstraction of GUI model
- GUI components are nice and modern
- Yahoo Map Flex component available
- Entire technology stack is not free (only SDK)
- Data Service may be costly
- Targets only Flash runtime environments

NetBeans and Eclipse Desktop Clients

Eclipse RCP

- Based on Eclipse IDE
- Desktop Application Framework
- Java-Based Plugin Model
- Built-In Updater
- GUI Based on Non-Standard SWT
- Uses Native Widgets
- Well-Designed Component Model
- Extended Event Model

Eclipse Considerations

- Runs on all popular OS's
- Great IDE support
- Large mindshare and community support
- Has support from numerous commercial and open source backers
- SWT is not standard and can present moderate learning curve
- GUI component model vastly different from standard Swing
- Expected native widget behaviors may differ

NetBeans Platform

- Based on NetBeans IDE
- Desktop Application Framework
- Java-based Plugin Model
- GUI Uses Standard Swing
- Built-in update mechanism

NetBeans Considerations

- Uses standard Java
- Great IDE Support
- Leverage knowledge of Swing
- Simple application component framework
- Increasing popularity and acceptance
- Support Java Web Start

Conclusion

Summary

- There are four main categories of rich clients to consider AJAX, XUL, Flash, and Desktop.
- Support for web-based rich clients is growing fast and there are plenty of tools available.
- Select a technology that best target your environment.
- AJAX-based solutions can target the largest user base since only a browser is required.
- Flash solutions are gaining popularity but should be tested first before adoption.
- IDE-based solutions are fat clients and should be examined carefully due to verbosity of code and Java knowledge required.

Resources

- AJAX
  - Rico - http://openrico.org/
  - Dojo - http://dojotoolkit.org/
  - Yahoo - http://developer.yahoo.com/
  - Google - http://code.google.com/
- XUL - http://developer.mozilla.org/
- OpelLaszlo - http://openloaszlo.org
- Adobe Flex - http://www.adobe.com/products/flex/
- Eclipse - http://www.eclipse.org/rcp/
- NetBeans Platform - http://www.netbeans.org/products/platform/