Building Next Generation Web 2.0 Applications

April 22nd, 2008
Introduction

Dion Hinchcliffe

- The Web 2.0 Blog
  - http://web2.socialcomputingmagazine.com
- ZDNet’s Enterprise Web 2.0
  - http://blogs.zdnet.com/Hinchcliffe
- Enterprise 2.0 TV Show
  - http://e2tvshow.com
- Hinchcliffe & Company
  - President and CTO
  - http://hinchcliffeandco.com
  - mailto:dion@hinchcliffeandco.com
  - Twitter: dhinchcliffe
- Web 2.0 University
  - http://web20university.com
Schedule

• From 1:00PM PDT-4:00PM PDT
• 5-10 minute Q&A at the end
What’s in this session?

- Overview of Web 2.0 applications, circa 2008
- Exploration of major design choices available to app developers now
- Discussion of pros and cons along the way
- Gain a deeper understanding of options and strategies
A Short History of Software

Popular Models for Developing and Integrating Software - 1970s to Now

- **1970s**: Structured Era
  - Code → operate → Data
  - Features, Subsystems

- **1980s**: Object-Oriented Era
  - Code → Data
  - Stateful distributed stateless
  - Subsystems, Systems

- **1990s**: Service-Oriented Era
  - Service → Data
  - Systems of Systems

- **2000s**: Web-Oriented Era
  - Resources → interact → Code

- **2010s**: Time (future)
Web 2.0 in a nutshell:

• “Networked applications that explicitly leverage network effects.” – Tim O’Reilly

• A network effect is when a good or service has more value the more than other people have it too.

• Two-way participation is the classic litmus test of a Web 2.0 system.
What is a Network Effect?

- A **network effect** occurs when a good or service has more value the more that other people have it too. - *Wikipedia*
  - Postal Mail
  - Phones
  - E-mail
  - Instant Messaging
  - Web pages
  - Blogs
  - *Anything that has an open network structure*
Triggering Exponential Growth

- Even small networks have large potential network effects.
- But very large networks have astronomical network effects.
- Recent Discovery: Reed’s Law, which say social networks are by far the most valuable.
Growth Pattern of Leading Web Sites vs. Average Sites

Legend:

1. Leading site that reaches organic growth “critical mass” and continues growing exponentially.

2. Average site that grows linearly and requires continuous efforts to grow and expand.
Network effects for exponential growth

Reselling its platform and data online netted $345MM in 2007.

Created the world’s leading online video service in 18 months with $10 million investment.

Built social networking platform into 60 million customer phenomenon worth $15-100B in 19 months.

Created the #1 satellite radio channel by handing over control of station 20 directly to their customers online.
Core Competencies of Web 2.0

• Online services, not packaged software, with cost-effective scalability
  – Non-trivially, this also includes 24x365 operations
• Maintaining control over unique, hard-to-recreate information that gets richer the more that people use it
• Trusting users as co-developers and co-creators
• Harnessing collective intelligence
• Exploiting lightweight business models
The Moving Pieces of Web 2.0

The Architecture of Web 2.0
“People in the Machine Nurture the Cloud”

People

Online Software Clients

Web 2.0 Site A Ajax Client
Web 2.0 Site B HTML Client
Web 2.0 Site C Flash Client
Mashup D Ajax Client
Mashup E Flash Client

Direct Web Servers & Services

Site A’s Web Services
Site B’s Web Server
Site C’s Web Services

Data

Database
Database
Web Service
Web Service
Web Service

Information “Cloud”

The “truthiness” barrier: Uncertain ownership and data provenance past this point
Network Effects + Harnessing Collective Intelligence =

Architecture of Participation:
Moving to the Center of Software Design

- User Contributions
  - Text, images, audio, video, news, etc.
- Database of Intentions
  - Searches, clicks, tagging, recommending, etc.
- Data Sharing
  - Prior similar uses, accumulated experience

“Collective Intelligence”

share

content

viral feedback loops

e-mail

invites

growth

exponential network effects

consume

audience

Moving to the Center of Software Design

http://web20university.com
Web 2.0 applied successfully: A pronounced “gravitational” effect
The Principles of Web 2.0

• The Web as Platform
• Harnessing Collective Intelligence
• Data as the next “Intel Inside”
• End of the Software Release Cycle
• Lightweight Software and Business Models
• Software Above the Level of a Single Device
• Rich User Experiences
• Innovation in Assembly
Integrating A Business Model
The most successful apps are fundamentally powered by data

- Potent datasets are behind all of the market leading products online today
- Information is the core value, more than software
- Value has moved “up the stack” from software to information
  - Driven by open source, cheap programming labor, and the incredible actual value that has accrued in data powered sites.
The Race Is On…

• To own major the classes of data online
  – Search, classifieds, product reviews, location, and many more

• Many classes of data already have established “owners”
  – Google, craigslist, Amazon, NAVTEQ, etc.

• **Another driver:** Why would someone use the second best source of data online when they can just use the best?
A map of opportunity…

• But there are many **classes of data online** that are still **unclaimed**
  – Or for which the established leaders have a fairly weak hold (i.e. social bookmarking)

These include:
  – Identity
  – Public calendaring of events
  – Parental control data
  – The good news: Hundreds more…
A Key Question:

• What is the best way to establish and maintain control of a hard to recreate set of data?

• The answer: The market leading firms online let their users primarily do it.
  – Getting “the clock” started on rapidly building the richest and most useful class of data.
Summary

“Web 1.0” Era
- Push business models
- Commercial software
- Customer service
- Bestseller products
- Traditional media
- 1-on-1 customer relationships
- Centralized product development

Web 2.0 Era
- Pull business models
- Open source software
- Customer self-service
- The Long Tail
- Social media
- Customer community relationships
- Decentralized product development

Unpredictability Volume
- Variety
- Output
- Peer production

Institutions
- Shift of control
- Central production

Communities of Individuals
But for us, Web 2.0 means all of this...
Today’s Web Applications Are Extremely Sophisticated

• Highly distributed and federated
• More social than ever before
• Built from cutting edge platforms and parts
• Have to scale to the Internet
• Expectations are high for functionality and low for the cost to develop new apps

Integrating with 3rd party suppliers live on the Web as well as being a 3rd party supplier is the name of the game circa-2008
It Also Means There’s A Lot To Master Today:
Part 1: Sourcing and Distribution

- 3rd party sourcing greatly accelerates development and provides rapid access to scalable, cost-effective infrastructure.
- New distribution models allow Web applications to have active presence across the Internet.
What’s being sourced and distributed?
What is 3\textsuperscript{rd} Party Sourcing?

- Using infrastructure, data, and functionality, usually live, in a Web application integrated from a 3\textsuperscript{rd} party over the Internet.
- Can be transparent to the user.
- Various business models exist but you will probably have to pay as you scale.
Advantages of 3rd Party Sourcing

- Leverage the economies of scale and operational competency of others
- Build on the shoulders of giants
  - Incorporate best-of-breed functionality in your app for incremental costs
- Improved Time-to-Market (10-20x factor)
Disadvantages of 3rd Party Sourcing

• Imposes reliability constraints
  – Weakest link syndrome
  – Amazon’s S3 was down for an extended period this year
• You have limited ability to leverage your own economies of scale
• Potential performance issues
  – i.e. 2nd network hop, perhaps more
• Requires as good legal skills as developer skills (SLA issues)
• Have to overcome Not-Invented Here
• Requires competency in assembly and integration as much as development
Common Methods of Sourcing

- All of the current distribution models available:
Drivers for Sourcing and Distribution: The Web has become a Platform

- Largest single global audience and marketplace
  - 1 billion user mark crossed in 2005
- The place where both software and data is moving
- Is the new über-platform
  - Older platforms such as radio, TV, newspapers, etc. are being pulled onto the Web wholesale
- Products and services of all descriptions are increasingly connected to the Web
- Walled gardens have come tumbling down
  - Being a single destination now greatly limits your success
Who is Providing Sourcing Today?

- Top 20 APIs ranked by use in mashups
- Many uses elsewhere as well

<table>
<thead>
<tr>
<th>API</th>
<th>Description</th>
<th>Category</th>
<th>Mashups</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google Maps</td>
<td>Mapping services</td>
<td>Mapping</td>
<td>1391</td>
</tr>
<tr>
<td>Flickr</td>
<td>Photo sharing service</td>
<td>Photos</td>
<td>329</td>
</tr>
<tr>
<td>YouTube</td>
<td>Video sharing and search</td>
<td>Video</td>
<td>245</td>
</tr>
<tr>
<td>Amazon eCommerce</td>
<td>Online retailer</td>
<td>Shopping</td>
<td>309</td>
</tr>
<tr>
<td>Microsoft Virtual Earth</td>
<td>Mapping services</td>
<td>Mapping</td>
<td>148</td>
</tr>
<tr>
<td>eBay</td>
<td>Online auction marketplace</td>
<td>Shopping</td>
<td>134</td>
</tr>
<tr>
<td>411Sync</td>
<td>SMS, WAP, and email messaging</td>
<td>Messaging</td>
<td>120</td>
</tr>
<tr>
<td>Yahoo Maps</td>
<td>Mapping services</td>
<td>Mapping</td>
<td>113</td>
</tr>
<tr>
<td>Google Search</td>
<td>Search services</td>
<td>Search</td>
<td>104</td>
</tr>
<tr>
<td>Yahoo Search</td>
<td>Search services</td>
<td>Search</td>
<td>104</td>
</tr>
<tr>
<td>delicious</td>
<td>Social bookmarking</td>
<td>Bookmarks</td>
<td>102</td>
</tr>
<tr>
<td>Yahoo Geocoding</td>
<td>Geocoding services</td>
<td>Mapping</td>
<td>73</td>
</tr>
<tr>
<td>Google Homepage</td>
<td>Portal gadgets</td>
<td>Widgets</td>
<td>55</td>
</tr>
<tr>
<td>Technorati</td>
<td>Blog search services</td>
<td>Blog Search</td>
<td>54</td>
</tr>
<tr>
<td>Last.fm</td>
<td>Music playlist management</td>
<td>Music</td>
<td>51</td>
</tr>
<tr>
<td>Facebook</td>
<td>Social networking management</td>
<td>Community</td>
<td>49</td>
</tr>
<tr>
<td>Twitter</td>
<td>Community site</td>
<td>Community</td>
<td>44</td>
</tr>
<tr>
<td>Google Ajax Search</td>
<td>Web search components</td>
<td>Search</td>
<td>43</td>
</tr>
<tr>
<td>Yahoo Image Search</td>
<td>Image search services</td>
<td>Search</td>
<td>39</td>
</tr>
<tr>
<td>GeoNames</td>
<td>Geographic name and postal code lookup</td>
<td>Mapping</td>
<td>37</td>
</tr>
</tbody>
</table>

Source: ProgrammableWeb.com
Questions about the Web Platform

• How does a vendor or product company gain advantage if the Web is a platform without an owner?
• Does creating your own platform within the Web platform create a walled-garden or a sustainable competitive advantage?
• Lesson: A platform tends to beat an application every time
The New Competitive Advantage Online: Creating a Compelling Platform Play
Example: Amazon.com

- **1st Gen. Product: E-commerce store**
  - No differentiation
  - Scaling of a single site
  - Single site

- **2nd Gen. Product: E-commerce platform**
  - 55,000 partners using their e-commerce APIs live
  - Scaling of the Web

- **3rd Gen. Product: A series of Web platforms**
  - Simple Storage Service (S3)
  - Elastic Compute Cloud (EC2)
  - Mechanical Turk (Mturk)
  - Many others
  - 300K businesses build on top of what they’ve produced

- **2nd and 3rd generation platforms generate large net revenue**
Successful 2nd Order Apps

JUNGLE DISK
Reliable online storage powered by Amazon S3™

Unlimited storage in the cloud for the cost of Amazon’s next megabyte

Many other examples available at:
http://amazon.com/aws

DigitalChalk
revolutionizing the learning experience

Provides an interface to create, deliver, and manage online training videos which is powered by Amazon S3, EC2, and SQS. DigitalChalk forecasts a first-year savings of over 75% by using AWS instead of a traditional data center.
Comparing Two of the Largest 3rd Party Sources
Another Critical 3rd Party Sourcing: Identity

- The Web has **no identity mechanism built-in**
- Users must identify themselves to each and every site via that site’s identity service
  - Different IDs and passwords everywhere
- **The most useful federated applications will need access to a user’s private data**
- Users are currently unlikely to hand over valuable credentials, such as the user id to their banking site, to 3rd party sites
- Addressing the identity challenge is “non-trivial” and poses challenges to making federated Web applications trustable by end users.
How Open Web Identity Works

• Web apps have to decide if their user accounts can be used on other sites and which open identity standard(s) they will accept.
• They also have to decide if they will accept other app’s accounts as valid logins.
• Users then only need to maintain one account, one password, and one set of credential, including reputation and other data.
Organizations are racing to solve Web identity

- Google Accounts
- Microsoft Live ID
- Sxip Identity 2.0
- OpenID.net
- Many others...

Google, IBM, Microsoft, VeriSign, and Yahoo!
A Key Web 2.0 Strategy: Turning Applications Into Platforms

• Openly exposing the features of software and data to customers, end-users, partners, and suppliers for reuse and remixing
• This strategy requires documenting, encouraging, and actively supporting the application as a platform
  – Has serious governance implications
• Provide legal, technical, and business reasons to enable this:
  – Fair licensing, pricing, & support models
  – A vast array of services that provide data that uses need
  – A way to apply these services to business problems rapidly and inexpensively.
New Distribution Models for the Web

- Spreading a Web application far beyond the boundaries of a site
- Becoming a 3rd party source to other Web applications via open APIs
- Being distributed as a visual widget, badge, gadget, SNS app, etc.
- Setting content loose via syndication to drive back traffic and drive network effects
Advantages of New Distribution Models

• Leverage other audiences and ecosystems other than your own
• Use the entire Web as a content and software distribution system
• Enable users to participate in product distribution 24x7 (lower marketing costs)
• Increase adoption and traffic
Disadvantages of New Distribution Models

- Unpredictable scaling and peaks
  - 90% of Twitter traffic is via the API

- Additional effort and expense required to develop and maintain each new distribution model

- Must maintain legal, technical, and community support infrastructure
Some important trends...

• The growth of Web sites with highly valuable “portable” content and functionality
• Users putting modular Web parts on their blogs and profiles to host the pieces of the Web that they want to share
  – By the millions on sites like MySpace
• The increasing realization that there is limited business value in being on a single site...
  – YouTube and Google showing the industry what’s possible here.
Further background

• “Atomization” of content is continuing
  – Small pieces are easier to reuse and more general purpose
  – Microformats are the smallest pieces
• The Do-It-Yourself trend is combining with the rise of Web portable content and functionality
  – People are helping themselves to what they like and building the experiences they can’t get any other way.
• The sophistication and scope of modern Web apps is driving the “sourcing” of functionality from existing scalable, online resources
  – Often representing major potions of Web functionality
The Web As a Parts Superstore

“There's little question that the Web is turning into a sort of online Home Depot with its shelves lined with thousands of useful, off-the-shelf parts of every description and utility.” – The Rise of the DIY Phenomenon, ZDNet
The “New” Philosophy of Web Distribution

• Jakob's Law states that "users spend most of their time on other people’s Web sites."

• So you must design your products and services to leverage this fact deeply in their design core.
Common Methods of Web App Distribution

• All of the current distribution models available:
Distribution Model #1: Open APIs

- The most common, in rough order:
  - RSS
  - REST
  - JSON
  - SOAP
How an Open API Creates a Platform

**Opportunity:**
*Going To the Customer and Open Web APIs*

Tens of Thousands of Dynamic Web Partners

---

**New Business Division:**
- Additional Indirect Revenue
- Direct Revenue

---

**Web 2.0 Site**

---

[http://web20university.com](http://web20university.com)
REST and Web-Oriented Architecture

- The most commonly used Web service approaches “in the wild” turn out to be the ones based on the “grain” of the Web:
  - **Representation State Transfer, or REST.**
    - Created by Roy Fielding, the co-creator of HTTP, the fundamental protocol of the Web.
    - Designed to fit naturally into Internet architecture
    - Extremely simple, not a standard, just a style of using HTTP
    - Fully embraces the workings of HTTP and uses its verbs (GET, PUT, POST, DELETE) on top of a granular, sensical URL structure to indicate what is to happen.
  - **RSS and ATOM**
    - ATOM is REST
  - **SOAP is not “bad”, but REST works better on the Web.**
Distribution Model #2: Widgets

- Turns your Web app’s functionality and content **portable**
  - Will work anywhere, not just your site
  - Be on millions of other sites instead of just your own.
- Users do the work of broaden your product’s distribution 24/7
  - Triggering **network effects** via **viral propagation**
- Widgets supply both mashups as well as hosting in blogs, wikis, & spaces
  - Turn your products into the foundation of other’s offerings on the Web
  - Gives many additional returns beyond the initial investment made in a Web app

**Crucial Point:**
Widgets Are Often Virally Self-Distributable, Triggering Network Effects
Widget Example: Google Gadgets
WidgetBox

Thousands of widgets for your blog, profile, or web page!

Find your widget...

Hotest widgets:
- Fun-Balloon
- Stamp Out Hunger
- Fundget
- Racism makes me sick
- Bubble Shoot

2,500,409,981 widgets served
564,515 domains served
50,295 widgets in gallery
SNS Apps

• Social networking sites are now allowing 3rd party Web apps to be embedded inside them.
• The space is in flux but very popular, with some SNS apps getting millions of users per day.
• Facebook has it’s own model. Google has theirs, OpenSocial. There are many others.
How OpenSocial Works

• Based on Google Gadgets
• Uses Atom Publishing Protocol to exchange data with the hosting OpenSocial container
• Based on very simple open standards
• Not an open source standard, yet
Building Facebook Applications on AWS

Amazon Web Services and Facebook are teaming up to help developers build instantly scalable applications for Facebook users. Build your Facebook app on AWS to ensure reliability, flexibility, and cost-effectiveness as your application grows in popularity.

With 55 million active members worldwide (as of November 2007), Facebook's social networking site allows you to create and share your applications with a massive and growing community of users. Amazon Web Services enables you to quickly implement your ideas for Facebook applications by providing an inexpensive, scalable computing platform. Using these infrastructure web services, your Facebook application is able to reach "web-scale" by scaling up and down seamlessly as demand dictates -- with pay-as-you-go pricing and no upfront costs.

To get started, simply sign up and start building:

1. Sign up for Amazon EC2 for elastic compute capacity in a virtual environment.
2. Sign up for a Facebook developer account.
3. Take advantage of Facebook and Amazon Web Services developer resources:
   - Hosting Facebook Applications on Amazon EC2 Tutorial
   - HelloWorld Facebook Application (AMI)
   - AWS Resource Center for plugins, command line tools, code samples and more.
   - AWS Simple Monthly Cost Calculator to estimate your monthly AWS costs.
4. Post your questions to the developer forums:
   - Amazon Web Services
   - Facebook
5. Launch your application on Facebook and apply for your Powered by Amazon Web Services logo.

App such as Booze Mail, iLike, Family Tree and SocialMoth Secrets were developed using this model.
Key Strategy: Enabling New Consumption Scenarios

- Cut-and-Paste deployment anywhere on the Intranet
- Consumption of a Web app’s data in any application that can use a URL
- Discovery of data via search
- Integration moves out of the spreadsheet
Definition: Mashup

• “A mashup is a Web site or Web application that seamlessly combines content from more than one source into an integrated experience.” - Wikipedia
• Content used in mashups is usually sourced from a 3rd party via a public interface (API)
• Other methods of sourcing content for mashups include Web feeds (e.g. RSS or Atom), and JavaScript/Flash “widgets”
Mashups

• Strong preference for reuse over coding
  – Innovation in assembly is the core value instead of ingenuity in coding
• Disruptive delivery model: Web-based with no install, no plug-ins, no admin rights, etc.
• Design focus is at the glue instead of the functionality
• Emphasis on simple, easy-to-use Web technologies over complex enterprise technologies
Mashup Examples

- HousingMaps.com
- ChicagoCrime.org
- Flashearth.com
- BidNearBy.com
- Pickaprof.com
- Tagbulb.com
Online Software with the Next Generation of Web Components

- Browser used for composition
- Informal yet effective integration models
- Zero footprint & lowadmin, with seamless upgrades
- Data from far flung federated Web services
- Blended Javascript applications
- Lightweight Web services (JSON, RSS, REST)

![Diagram showing components of web software]

- Web service proxy
- Web service
- Web Server
- Database
- Ajax Application
- Browser
- Javascript Framework
- Ajax Library
- Ajax Widget
Mashups Move Towards Standards To Create a Consistent Canvas
Mashup Styles

- Simple formats and standards
- Can “mash-up” at the server as well as the browser.
- Mashups can take place at the presentation, data, or code level.
The major approaches to mashup development

• Developer-Oriented
  – Ajax frameworks and libraries
  – Live Javascript includes
  – Flash/Silverlight components

• Prosumer-Oriented
  – Mashups Tools (JackBe Presto, Yahoo! Pipes)
  – Enterprise 2.0 platforms

• Consumer-Oriented
  – Cut-and-Paste widgets, badges, gadgets
The Rise of Ajax Has Helped Drive Mashups and Widgets
Part 2: Rich User Experiences

• The move away from static Web pages to dynamic experiences that feel like native software.
• Major software vendors such as Adobe, Microsoft, and Sun are entering the space.
• Result: The Web becomes a true software platform
Benefits of RIA

• RIAs offer a richer interface that provides a more engaging user experience without the need for page reloads.
• RIAs offer real-time feedback and validation to the user, triggered by user events
• The look and feel of a traditional desktop application can be accomplished with a RIA
• RIAs also can include a full multimedia experience, including audio and video
• RIAs have capabilities such as real-time chat and collaboration that are either very difficult or simply impossible with traditional Web applications.
Ajax: Asynchronous Javascript and XML
Overview

Ajax
The organic and 100% open standards-based RIA model
Overview of Ajax

• Coined as a term in February, 2005 by Jesse James Garrett of Adaptive Path.
  – However, really a technique used as far back as 1998.
• But came into vogue particularly with the power of new browsers, Javascript and XmlHttpRequest().
  – And the rise of fast connections to the Web.
• Concept: The Web page hosts entire Javascript programs
  – The UI is manipulated programmatically and in real-time by changing the Document Object Model
  – The Web page isn’t reloaded unless completely new functionality is needed.
  – Information is accessed in the background (asynchronously) by the browser via Web services
    • Data is in XML or JSON format. Or else anything HTTP(S) can transmit.
The Result…

• **Pure browser software** that can be every bit as rich as native PC software.
• The Web becomes a **true software platform** in its own right.
• **An open software model** that has no owner.
  – Not vendor controlled, just based on the neutral, free standards of the Web.
• **A significant challenge** as the browser client suddenly becomes quite complex.
The Next Generation of Web Applications:
Ajax - Dynamic User Interface, SOA Enabled, OS/Browser Independence

DOM
Application Code (JavaScript)
Ajax Framework (JavaScript)
XML Data
Ajax Application Client

SOA Environment

SOAP Service
REST Service
JSON/HTTP Service
POX/HTTP Service
WCF (Indigo) Service
WS-* Service

XMLHttpRequest (Asynchronous)

Copyright (c) 2005 by Dion Hinchcliffe
More Sophisticated View of Ajax
Ajax for Software Composition

Online Software with the Next Generation of Web Components

- Browser used for composition
- Informal yet effective integration models
- Zero footprint & low admin, with seamless upgrades
- Data from far flung federated Web services
- Blended Javascript applications
- Lightweight Web services (JSON, RSS, REST)

Diagram:
- Ajax Library
- Javascript Framework
- Ajax Widget
- Ajax Application
- Web service proxy
- Web service
- Database
- Web Server
- to external Web service
Example Ajax Architecture
Key Aspects of Ajax

- Uses browser abilities never originally intended for building Rich Internet Applications
- Based on 100% open standards
- Cannot support rich media (audio, video)
- Has poor visual assembly and declarative support overall (a few exceptions from major vendors)
- Can run on any browser, including mobile and embedded devices that support Javascript, DHTML, and XmlHttpRequest
- Example Applications: LZPix
Microsoft Silverlight: Formerly Windows Presentation Foundation/Everywhere
Overview

Microsoft Silverlight

A comprehensive RIA platform that pushes the client to the next level

Browser

- XAML defined visual object model
- User events
- DOM Manipulation

Common Language Runtime
- JavaScript
- Python
- Ruby

Initial App Load
- Async Data Calls
- Web Services (REST, JSON, SOAP)

HTTP Server

Web Server

hinchcliffe & company
The future of business, now.

Web 2.0 University
http://web20university.com
Silverlight Basics

- A cross-browser, cross-platform downloadable plug-in from Microsoft.
- Delivers a next generation of .NET based media experiences and rich interactive applications (RIAs)
- Offers a flexible and consistent programming model that supports AJAX, Python, Ruby, and .NET languages such as VB and C#
- Commercial but has some open standards support including Javascript.
- Has powerful rich media capabilities including fast, cost-effective delivery of high-quality audio and video to all major browsers including Firefox, Safari and Internet Explorer running on the Mac or on Windows.
- Primary tools are Expression Studio and Visual Studio,
  - Designed for manufacturing: Designers and developers can collaborate effectively to create Silverlight web experiences.
- Example: The Silverlight Showcase and Intersoft Solutions
Basic Concepts

• Silverlight development is similar to HTML development
  – Presentation format for Silverlight is XAML
    • XAML is XML (markup)
  – Silverlight V1.0 programming language is JavaScript
• AJAX tricks work with Silverlight
• Minimum tools are a text editor and a web browser
Initial Platform Elements

• Key Components in Silverlight v1.0
  – 2D Graphics (shapes, transformations, clipping, masking)
  – Animations
  – Input: Mouse and Keyboard
  – Media (WMV, WMA, MP3)
  – Imaging (JPG, PNG)
  – Text (static and flow)
  – HTTP downloader
  – XAML Parser (includes CreateFromXAML API)
  – JavaScript DOM
Silverlight: Comparing XAML and Javascript

<Canvas xmlns="http://schemas.microsoft.com/client/2007"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    MouseEnter="OnMouseEnter">
    <TextBlock Canvas.Top="30" Foreground="#FFFF3333">
        This is some text
    </TextBlock>
</Canvas>

Canvas canvas = new Canvas();
canvas.MouseEnter += new MouseEventHandler(OnMouseEnter);

TextBlock t = new TextBlock();
t.SetValue(Canvas.TopProperty, 30);
t.Text = "This is some text";
Color ratherRed = Color.FromArgb(0xFF, 0xFF, 0x33, 0x33);
t.Foreground = new SolidColorBrush(ratherRed);
canvas.Children.Add(t);
Microsoft’s UX Model

- User touch points: Windows, Office, Web, Devices, Digital home
- Experience:
  - Web Standards: ASP.NET, AJAX
  - Enhanced browser: Silverlight
  - Windows: .NET Framework 3.0
- Presentation Model:
  - CSS/DHTML
- XAML
- Programming Model:
  - HTML + AJAX + ASP.NET
  - .NET Framework
Silverlight’s Place in MS App Strategy

• **Non-Proprietary and Open Apps, Non-Rich Media**
  - ASP.NET AJAX for AJAX-enabled applications built around Web standards. ASP .NET AJAX allows standard Web applications to be more effective by improving the interaction parameters of the application (such as refresh, resource usage, and navigation).

• **Multiplatform Rich Media Apps**
  - For Web experiences requiring 2-D animation, vector graphics, and high-fidelity audio and video on the Web, the Silverlight plug-in renders XAML in addition to HTML.
  - Embraces Web architecture for development, including industry-standard AJAX (Asynchronous JavaScript+XML) and inline XML markup (XAML) for presentation,
  - Content authored in Silverlight and ASP.NET AJAX becomes more discoverable while offering the benefits of being cross-platform (Windows and Macintosh) and cross-browser (Internet Explorer, Firefox, Safari).

• **Traditional Native Windows Apps That are Connected**
  - Use .NET Framework 3.0 programming layer (shipped in Windows Vista and available for Windows XP) that includes the Windows Presentation Foundation (WPF).
  - WPF can create rich, immersive, connected applications and experiences that can take advantage of the Windows platform, including UI, media, offline communication, and document support.
    - WPF uses a superset of the same XAML that is used by Silverlight.
Result: The UX App Strategy

- Unified development environment
- Best of breed capability (3d graphics, high def video, extreme ease of development)
- Limited standards support
- Backed by the largest software company in the world.
The four way to use Silverlight
Two Silverlight Development Models
Text Editor or IDE
And two versions of Silverlight

• v1.0 = Basic plug-in + XAML rendering

• v2.0 =
  – v1.0
  – + managed code (CLR)
  – + XAML extensibility
  – + Control class (user control)
  – + sample controls
The full Silverlight vision is extensive.
Intended Uses of Silverlight

- Web media— Branded playback with events, video and marketing mix, dynamic videos with ads, audio playback, and so forth
- Rich islands on a page (mini apps)— Casual games and gadgets
- Web visualization elements— Navigation properties, data visualization, and ads
- Web applications – Business and personal productivity offered as software-as-a-service
Sun’s JavaFX: Declarative Web Apps That Leverage the Java Platform
Overview

JavaFX

A declarative RIA platform that leverages the Java platform with best practices and high efficiency

Source: http://hinchcliffe.org
Introducing JavaFX Script and Mobile

• JavaFX Script: This new scripting language gives Java developers the ability to quickly create content-rich applications for the widest variety of clients
  – Includes mobile devices, set-top boxes, desktops, and Blu-ray disc players.
  – Content creators are given a simple way to develop content for any Java–based consumer device.

• JavaFX Mobile: A complete software system for mobile devices. It is available via OEM licenses to carriers, handset manufacturers, and other companies.

• Example: The WeatherFX application
JavaFX Capabilities

- Dynamic Interactive Content: Delivered on most existing Java-powered devices.
- Write-Once Run Anywhere: Unlike technologies like Ajax.
- Integration and Ease of Use: JavaFX allows content creators to create rich media content without relying on developers, including drag and drop of desktop and mobile content to the desktop, something that is not possible in any other RIA.
- Leveraging the Java Platform: JavaFX Script offers a close integration with other Java components (applications and infrastructure) running on server and client platforms, enabling a rich end-to-end experience for developers and users.
- Secure Digital Asset Access: JavaFX Script takes advantage of the Java security model so consumers can securely access assets (e.g., pictures, music files, word documents) on their desktop.
JavaFX Roadmap
Key concepts of JavaFX

- JavaFX Script is a scripted language, not compiled
- Can make use of the entire Java run-time and existing Java libraries
- Designed to make development of RIAs on all Java platforms easier than it is today
- Does not necessarily replace libraries like Swing, rather JavaFX makes Swing easier to use
- JavaFX applications can run in the browser, on the desktop, and in mobile devices
- JavaFX is explicitly designed to create arbitrary large applications, it has essentially the same restrictions as Java
Value-Proposition

- Scripting language that is more efficient and more closely matches the layout of a Web applications
- Will be an open source solution
  - Not at the moment however
- Specifically designed for today’s Rich Internet Applications (unlike Ajax)
- Leverages the vast reach of Java on desktop, mobile, and embedded devices
- Best practices including MVC and support for Web design integration
- Part of a new and comprehensive evolving strategy to move Java to the Web
  - Such as the freshly open sourced Project Scene Graph
Adobe Flex & AIR
Key Capabilities

• Built on the pervasive Flash platform
• Simple declarative programming model
• Extremely good development tool support
• Based on largely open standards (except Flash)
• Increasingly Web-oriented
• Has potent capability like pub/sub, real-time data push, round-trip debugging and more.
RIA Challenges

- Searchability of RIA content
- Page Views
- Accessibility
- Analytics Support
- Testing
- XML support can be troublesome in Flash
- Others
Part 3: New Development Platforms

• The Web development industry is moving into a focus on “productivity-orientation.”
• New platforms highly optimized for Web development are emerging.
• These new Web development platforms embody much of what we’ve learned in the last 15 years in terms of best practices.
• However, like all platforms, they have tradeoffs, including performance and maturity.
Performance of New Languages Is An Issue

**Hard Metrics:**
The run-time performance of various programming languages across 17 benchmarks

- **Data Source:** http://shootout.alioth.debian.org

**21st Century Web Development:**
The classic struggle between developer productivity and run-time efficiency

- **C/C++**
  - run-time performance
- **Python**
  - developer productivity
- **Java**
- **Assembly**
- **.NET**
- **Perl**
- **Ruby on Rails**
- **Tcl**
- **PHP**
- **CakePHP**
- **JavaScript**

**Note:** “Platforms” here means the language plus any run-time libraries/frameworks.
Ruby on Rails

- Ajax-ready but works with all RIA technologies
- Automatic Object/Relational Mapping
- Sophisticated Model View Controller Support
- Convention over configuration
- Radically-oriented around Web development only
- Very high productivity (IBM verified 10-20x older platforms)
- Open source and free
- Runs major sites like Twitter
- One of the most popular new platforms
- Has clones in most other major languages now
CakePHP

- Open source Web application framework written in PHP
- Works with all major RIA technologies
- Modeled after the concepts of Ruby on Rails
- Not a port of Rails but extends the ideas to PHP
- Stable, mature, and reliable
Groovy & Grails

- Groovy is a dynamic language for the Java Virtual Machine
- Has strengths of Ruby, Python, and Smalltalk
- Runs anywhere Java runs
- Grails is a Ruby on Rails like framework for Groovy
- Mature, stable, and relatively high performance
Platform-as-a-Service (PaaS)

- Increasingly, everything is being integrated together and offered as a platform as a network service:
  - Google App Engine
  - Amazon’s AWS
  - Bungee Connect
  - Heroku
  - Elastic Server On Demand
Recommendation: Keep Sight of the Goals

• First, learn the Web business.
• Spend time studying the competition.
• Really get to know your customers.
• Along the way, don't lose sight of the fundamentals of Web 2.0.
• Finally, use all the latest tools, technologies, apps, platforms and gain ground truth on what they can do.
Conclusion

• An enormous range of choices make developing next-generation Web apps easier than ever, yet there is more than ever to learn.
• Very stiff competition on the Web means leveraging all the strengths you can find, anywhere you can find them.
• There is no requirement to adopt every new technology and technique, however.
  – Pilots and proof-of-concepts can teach you a lot
  – Simplicity and pragmatism tends to win at the end of the day on the Web.
Questions?

Slides:

dion@hinchcliffeandco.com