Today, many projects are increasingly focused on web based applications using web browsers as their primary user interface. How durable is this application architecture? Is the browser likely to continue to expand its primacy? Can we expect the basic structure of our web based applications to remain fairly stable for the foreseeable future or do we need to be preparing to make drastic changes? If the browser is a transitional technology, what might replace it? This talk will explore these and related issues and help you understand how application development is about to change.
A little background about me

- Compilers, Smalltalk virtual machines, GCs, language design, development tools
- Launched first commercial Smalltalk: Tektronix 4404
- Digitalk/Parcplace-Digitalk: Enterprise Scale Smalltalk
- (Re-) Instantiations: JOVE Java optimizing compiler, Eclipse tools
- Microsoft: JavaScript/ECMAScript 5
- Mozilla: Future of JavaScript and the Web platform

http://youtu.be/8yxCjhayW-8
Are We Really Living in a Post-PC World?

By Jason Cross, PCWorld  Mar 4, 2011 1:30 PM

Among the many ear-catching bon mots issued forth from Steve Jobs recently is the assertion that we're living in, or at the very least entering, the post-PC era. It started last year at the D8 conference, when the Apple CEO said that PCs are going to be "like trucks" in that they'll still be around and useful for certain work, but only a smaller percentage of the users will need one. More recently at the iPad 2 launch, Jobs reconfirmed this post-PC mantra.

VMware boss focuses on post-PC era at VMworld

By: Jay Groone  AUGUST 29, 2011 6:41 PM PDT
Hardware evolution doesn’t define a computing era.

Computing eras are about the impact of computing on society.
The Eras of Computing

1950
1960
1970
1980
1990
2000
2010
2020

Computers empower/enhance
individuals’ tasks

Computers empower/enhance
enterprise tasks

Corporate Computing

Personal Computing

A New Era?

Societal Impact

Computers empower/enhance
individuals’ tasks

Computers empower/enhance
enterprise tasks
If we are entering a new era of computing, what is it?

• The Post-PC Era?
• The Mobile Era?
• The Cloud Era?
• The Tablet Era?
• The Games and Media Player Era?
• The Monetize the User Era?
The Ambient Computing Era

- Devices not Computers
- Ubiquitous access to information
- Computing augmented life

Computers enhance the world I live in. I need my stuff (data and apps) right now, wherever I am, using whatever device is available. I can’t live without it!
Transitional Technologies

• Precursors of fundamental change
• Exhibit some characteristics of what will come
• Firmly rooted in the status quo
Transitional Technologies
Precursors of PC Era

- Minicomputers
- Timesharing

Societal Impact


Personal Computing
Corporate Computing
Transitional Technologies
Precursors of Ambient Computing Era

- **1950**
- **1960**
- **1970**
- **1980**
- **1990**
- **2000**
- **2010**
- **2020**

**Societal Impact**
- **Minicomputers**
- **Timesharing**
- **WWW/Browsers**
- **Cell phones**
- **Ambient Computing**
- **Personal Computing**
- **Corporate Computing**

Transitional Technologies
Precursors of Ambient Computing Era

- **1950**
- **1960**
- **1970**
- **1980**
- **1990**
- **2000**
- **2010**
- **2020**

**Societal Impact**
- **Minicomputers**
- **Timesharing**
- **WWW/Browsers**
- **Cell phones**
- **Ambient Computing**
- **Personal Computing**
- **Corporate Computing**
The Cell Phone

- Always with you
- Always connected
- But initially dedicated to a single use case
- Initially not very smart
The WWW/Web Browser

- First real ubiquitous computer augmented availability of unstructured information
- Universal client/server computation model (HTTP/REST)
- Universal presentation platform
- Availability more important than performance
- A Web Browser is a classic PC application
Why are these only transitional

• Telephony is just a single application
• Wireless communications now common part of most devices
• WWW data access is an essential part of most applications
• Users don’t need a “browser” to access the web any more
• Growing focus on solutions (apps) not tools (the browser)
• When something becomes ubiquitous, it disappears
Every Computing Era Has a Dominant Application Platform

- Corporate Computing Era: IBM Mainframes
- Personal Computing Era: Microsoft/Intel PC
- Ambient Computing Era: T.B.D (or is it?)

Created by Market Demand, “Good Enough” Technical Foundation, and Superior Business Execution
What do you have when you strip away the PC application part of a web browser?
What do you have when you strip away the PC application part of a web browser?

Frameworks and Libraries

Language Runtime

The Web Client Application Platform

HTML
CSS
JavaScript
SVG

Styling
Layout
Rendering

Network
Local storage
User Input
The Ambient Applications Platform
Web Apps vs. “Apps”

- HTML/CSS/JavaScript/WebApp APIs
- Server via: HTTP, JSON, XML, Sockets
- Works on any device
- Task specific “Chromeless” UIs emerging
- “App Stores” about to emerge

- Proprietary Language+Frameworks (Java, Objective-C, C#, etc.)
- Server via: HTTP, JSON, XML, Sockets
- Works only within a device family
- Task specific “Chromeless” UIs
- Tied to a specific “App Store”
New Platform Emergence

Other App Dev Technology Stacks

Desktop Manager
Desktop services and frameworks
Graphics services and frameworks
OS Display Abstractions
Kernal
Device Drivers

Frameworks and Libraries
- HTML
- CSS
- JavaScript
- SVG

Language Runtime
- Styling
- Layout
- Rendering
- Network
- Local storage
- User Input

“Modern” OS Circa 2005
New Platform Emergence

- Mozilla Boot to Gecko
- Chrome OS
- Palm/HP WebOS
- Etc.

Ambient Device OS Circa 2012
Multitouch

Accelerometer

Cameras

Speaker

NFC

Vibration Motor

USB Access

Microphone

Light Sensor

Bluetooth

Proximity Sensor

Gyro

Hardware Keys

Light Sensor

Proximity Sensor

Gray = Web APIs Exist

Yellow = Web APIs still needed
Areas of real concern

- Is it possible for a vendor neutral standards-based open platform to achieve similar dominance?
- Cumbersome standardization processes
  - Really! W3C, Ecma, WhatWG, IETF, Kronos, …
- Who really drives innovation?
- Too slow rate of change?
- What actually gets implemented?
Each Computing Era has had Canonical Programming Languages

- Corporate Computing Era – COBOL/Fortran
- Personal Computing Era – C/C++ family
- Ambient Computing Era – JavaScript ??
Why JavaScript?

Because “Worse is Better”

• It’s there – It’s working
• It’s good enough
• It’s getting better
• What could replace it?
• How could that happen?

Dick Gabriel
http://www.dreamsongs.com/WorseIsBetter.html

JavaScript Performance Over Time

5 years of progress...

one program on one popular browser:
10x faster!


http://iql2.com/blog/as3-benchmark/
PDF Renderer in JavaScript

Hence, recording and compiling a trace speculates that the path and typing will be exactly as they were during recording for subsequent iterations of the loop.

Every compiled trace contains all the guards (checks) required to validate the speculation. If one of the guards fails (if control flow is different, or a value of a different type is generated), the trace exits. If an exit becomes hot, the VM can record a branch trace starting at the exit to cover the new path. In this way, the VM records a trace tree covering all the hot paths through the loop.

Nested loops can be difficult to optimize for tracing VMs. In a naïve implementation, inner loops would become hot first, and the VM would start tracing there. When the inner loop exits, the VM would detect that a different branch was taken. The VM would try to record a branch trace, and find that the trace reaches not the inner loop header, but the outer loop header. At this point, the VM could continue tracing until it reaches the inner loop header again, thus tracing the outer loop inside a trace tree for the inner loop. But this requires tracing a copy of the outer loop for every side exit and type combination in the inner loop. In essence, this is a form of unintended tail duplication, which can easily overflow the code cache. Alternatively, the VM could simply stop tracing, and give up on ever tracing outer loops.

We solve the nested loop problem by recording nested trace segments as they are encountered, recording a branch trace of the type not taken, and compiling the new portion of the trace on entry. This avoids the issues caused by recording a branch of the outer loop.
JavaScript Performance:

http://mbebenita.github.com/Broadway/broadway.html

Emscripten

Compiling C/C++ to JavaScript

This is CPython, the standard Python implementation, compiled from C to JavaScript using Emscripten, running in your browser (without any plugins).

- Most core language stuff should work, except for importing non-static modules (in other words, `import sys` will work, but other modules won't).
- Please report bugs if you find them!
- Tested on Firefox 4 and Chrome 10.
- The editor is Skywriter.

Enter some Python:

```python
import sys
print 'Hello world! This is Python {} on {}'.format(sys.version, sys.platform)
print 'Here are some numbers:', [2*x for x in range(5)][4]
```

Hello world! This is Python 2.7.1 (r271:06832, Dec 12 2010, 15:26:43)
GCC 4.2.1 (Based on Apple Inc. build 5658) (LLVM build) on linux2

Here are some numbers:
Langauges That Compile To JavaScript

- CoffeeScript ("Improved" JavaScript syntax)
- Java (GWT)
- Script# (C# dialect)
- Dart
- ClojureScript (Lisp dialect)
- Dozens more listed at https://github.com/jashkenas/coffee-script/wiki/List-of-languages-that-compile-to-JS

JavaScript is the “virtual machine” of the Web Platform.
Evolving JavaScript
The ECMAScript Standards Process

Ecma International Technical Committee 39 (TC39)

ES 3 (1999)
ES 5 (2009)
ES 5.1 (2011)
“ES 6” (2013)
“ES 7” (2017?)
...

“ES.next”
“Harmony”

E4X
“ES4”

mozilla
Evolving JavaScript

The ECMAScript Standards Process

Ecma International Technical Committee 39 (TC39)

ES 3
(1999)

ES 5
(2009)

ES 5.1
(2011)

“ES 6”
(2013)

“ES 7”
(2017?)

…

- Classic ISO targeted “pay to play” standards organization
- Open public discussion list es-discuss@mozilla.org
- Working documents all publically available at wiki.ecmascript.org
Interoperability is TC39's highest priority.

- A detailed and highly prescriptive specification
- Large, non-normative test suite for implementers

http://test262.ecmascript.org/
Evolving JavaScript
ECMAScript Harmony Goals

1. Be a better language for writing:
   A. complex applications;
   B. libraries (including the DOM) shared by those applications;
   C. code generators targeting the new edition.

2. …

http://wiki.ecmascript.org/doku.php?id=harmony:harmony
Major ES.next Enhancements

- Modules
- Sanding-boxing module loaders
- Proxy Objects – low level behavioral intercession
- Control abstraction via iterators and generators
- Array comprehensions, String interpolation
- super references
- Encapsulated state via gensym-like private names
- Better support for class-style inheritance
What about the server?

• The server-side of applications don’t have the “single platform” imperative of clients: Java, .Net, Python, Ruby

• Growing interest in also using JavaScript for the server portion of applications

Developers want to use same skills and technical resources on both client and server

• node.js ([http://nodejs.org/](http://nodejs.org/)) has exploded on the scene

```javascript
//hello world server
var http = require('http');
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/plain'});
  res.end('Hello World
');
}).listen(1337, "127.0.0.1");
console.log('Server running at http://127.0.0.1:1337/');
```
What About “The Cloud”
Concurrency/Using Multicores

• The Browser App Platform uses a very simple no shared state concurrency model.
  § Web Workers
  § PostMessage and JSON data records
  § Event loops / Continuation passing style

• Essentially the same model as browser ↔️️ webserver

• Can be fully distributed, not just client/server.
Clouds All the Way Down?

- Common programming models
- Common APIs
- Local/remote cloud transparency
Proprietary Silos or a Standards-based Open Platform?

Walled Gardens
Users: Customers or Merchandise?

• Are we building devices and services that empower people?
• Or, are we just trying to attract their attention so we can turn them into monetizable assets?
User Sovereignty

• Can we keep the user in control of their computer augmented life?

• Some of the key issues
  - Identity
  - Privacy
  - Freedom to use what we choose
  - Reliable storage of our digital assets
  - Durability of data and applications

“Mozilla’s mission is to build user sovereignty into the fabric of the internet”

Mitchell Baker
Mozilla Chairperson
What’s It All Mean?

• A twenty year vision is easy to craft, the path that will take us there is unknowable.

• The life of an innovator at the beginning of an era is very different from the waning days of an established era.

• The browser derived web client platform is a strong candidate to be the dominant application platform of the Ambient Computing Era.

• If you aren’t all ready doing so, it’s time to take JavaScript seriously.

• We are all users, let’s create the world we want to live in
This is the most exciting time for the computing industry since the 1980’s.

Enjoy it!
Follow up


allenwb@mozilla.com

@awbjs