Computational REST: A new model for Decentralized, Internet-Scale Applications

Justin R. Erenkrantz

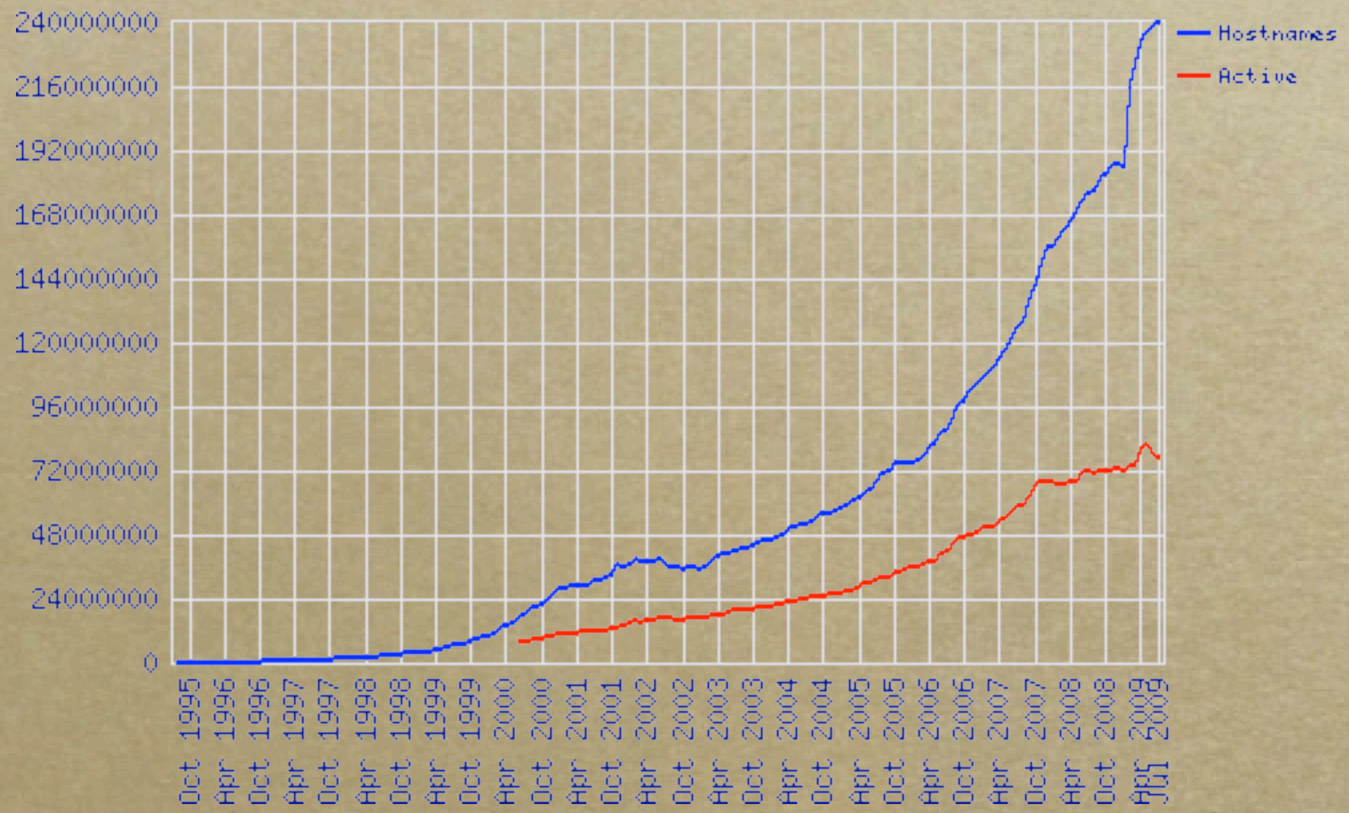
Final Defense

September 3, 2009

This material is based upon work supported by the National Science Foundation under Grant Numbers 0438996 and 0820222. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

Runaway success of the Web

http://www.worldwidewebsize.com/: *21.8 billion* *pages*



Netcraft Site Count History

http://news.netcraft.com/archives/2009/07/site_count_history.png

Netcraft

239 million sites

30% are "active"

July 2009

The Nielsen Company

Average home user views

1,591 pages/month

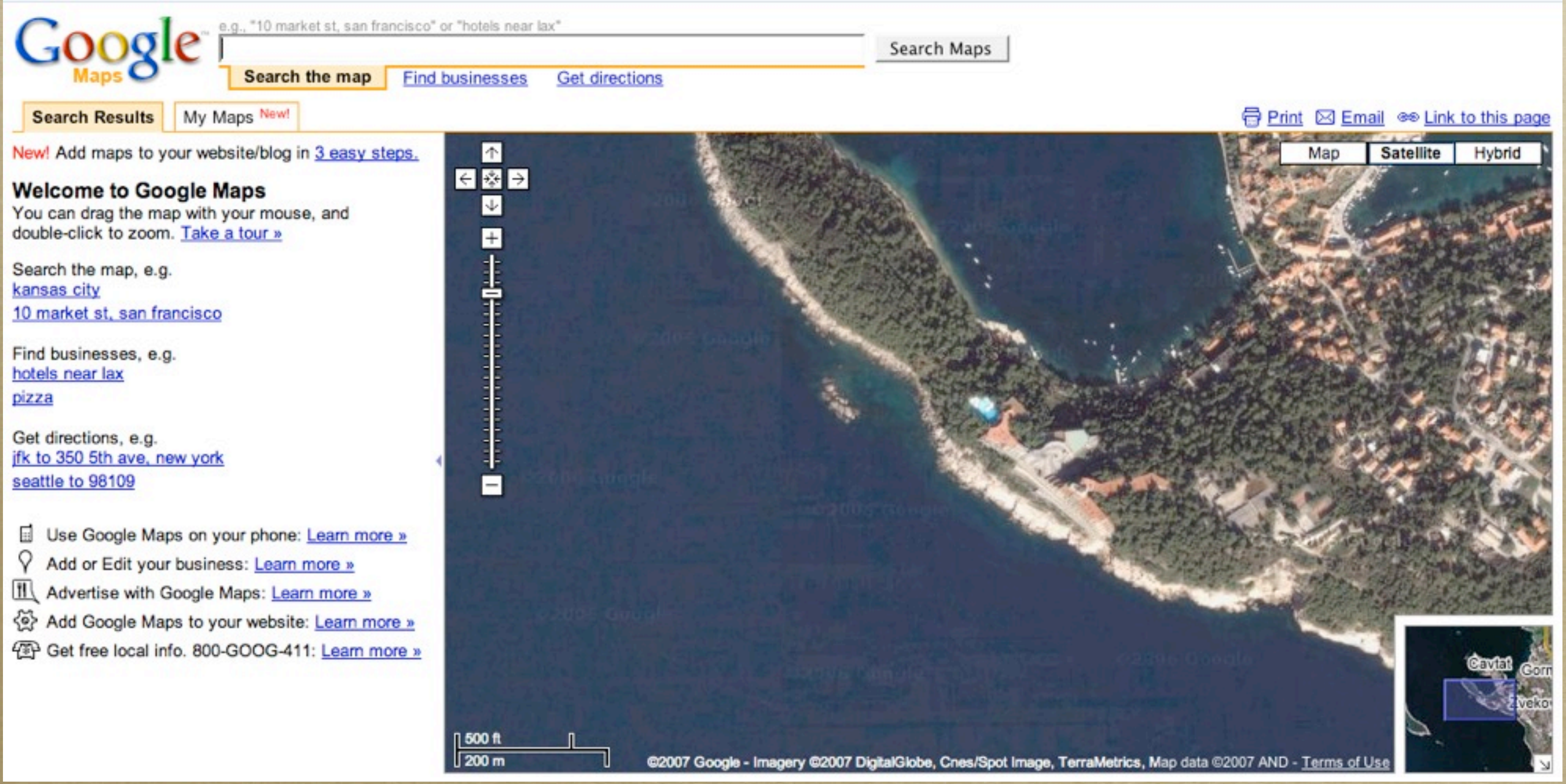
May 2009

http://www.nielsen-online.com/resources.jsp?section=pr_netv

How did we get here?

- *In the early-to-mid-90s, the Web faced a crossroads: how could the Web scale?*
- *Software architecture, in particular, the REpresentation State Transfer (REST) style [Fielding, 2000] guided crucial reformations introduced in HTTP/1.1*
- *Permitted the superscaling of the Web*

Puzzling web apps: Google Maps



New! Add maps to your website/blog in 3 easy steps.

Welcome to Google Maps
You can drag the map with your mouse, and double-click to zoom. Take a tour »

Search the map, e.g.
kansas city
10 market st, san francisco

Find businesses, e.g.
hotels near lax
pizza

Get directions, e.g.
jfk to 350 5th ave, new york
seattle to 98109

Use Google Maps on your phone: Learn more »
Add or Edit your business: Learn more »
Advertise with Google Maps: Learn more »
Add Google Maps to your website: Learn more »
Get free local info. 800-GOOG-411: Learn more »

©2007 Google - Imagery ©2007 DigitalGlobe, Cnes/Spot Image, TerraMetrics, Map data ©2007 AND - Terms of Use

Research Question

What happens when dynamism is introduced into the Web?

Dynamism: Phenomena that must be explained as a manifestation of change, whether through interpretation or alteration of the interpreter.

Dramatis personae of the Web

- *REST: architectural style*
- *HTTP/1.1: protocol governed by REST*
- *URI: naming convention (http://...)*
- *Apache HTTP Server: origin server*
- *Squid: gateway and proxy*
- *Firefox, Safari, Internet Explorer: user agent*

REpresentation State Transfer

- *Started in mid ‘90s; captured in Fielding’s dissertation (2000) and TOIT (2002)*
- *Guided the reformations introduced in HTTP/1.1 and URI specifications*
- *Designed for Internet-scale distributed hypermedia*
- *Few (if any) clarifications since then…*

REST Axioms

1. *The key abstraction of information is a resource, named by an URL.*
2. *The representation of a resource is a sequence of bytes, plus representation metadata to describe those bytes.*
3. *All interactions are context-free.*
4. *Only a few primitive operations are available.*
5. *Idempotent operations and representation metadata are encouraged in support of caching.*
6. *The presence of intermediaries is promoted.*

Approach

- *Examine evolution of key infrastructure applications*
- *Identify the root causes of dissonance in Web applications which are not fully explained by REST*
- *Introduce a named set of architectural principles (style) that provide more applicable guidance for web applications*
- *Assess the effectiveness of these principles by:*
  - *characterizing the dissonance seen in existing systems against new style*
  - *creating a framework based on the new style that demonstrates novel web applications*

Insight: Apache modules & Browser plugins

With little explicit coordination among developers during this period, critical web infrastructure applications evolved rapidly to support dynamism - both architectural and content-focused.

Insight: mod_mbox/Subversion

Even for knowledgeable practitioners of REST, REST, in isolation, does not provide enough design guidance for architects to understand why applications fall into architectural dissonance.

Insight: Web Services

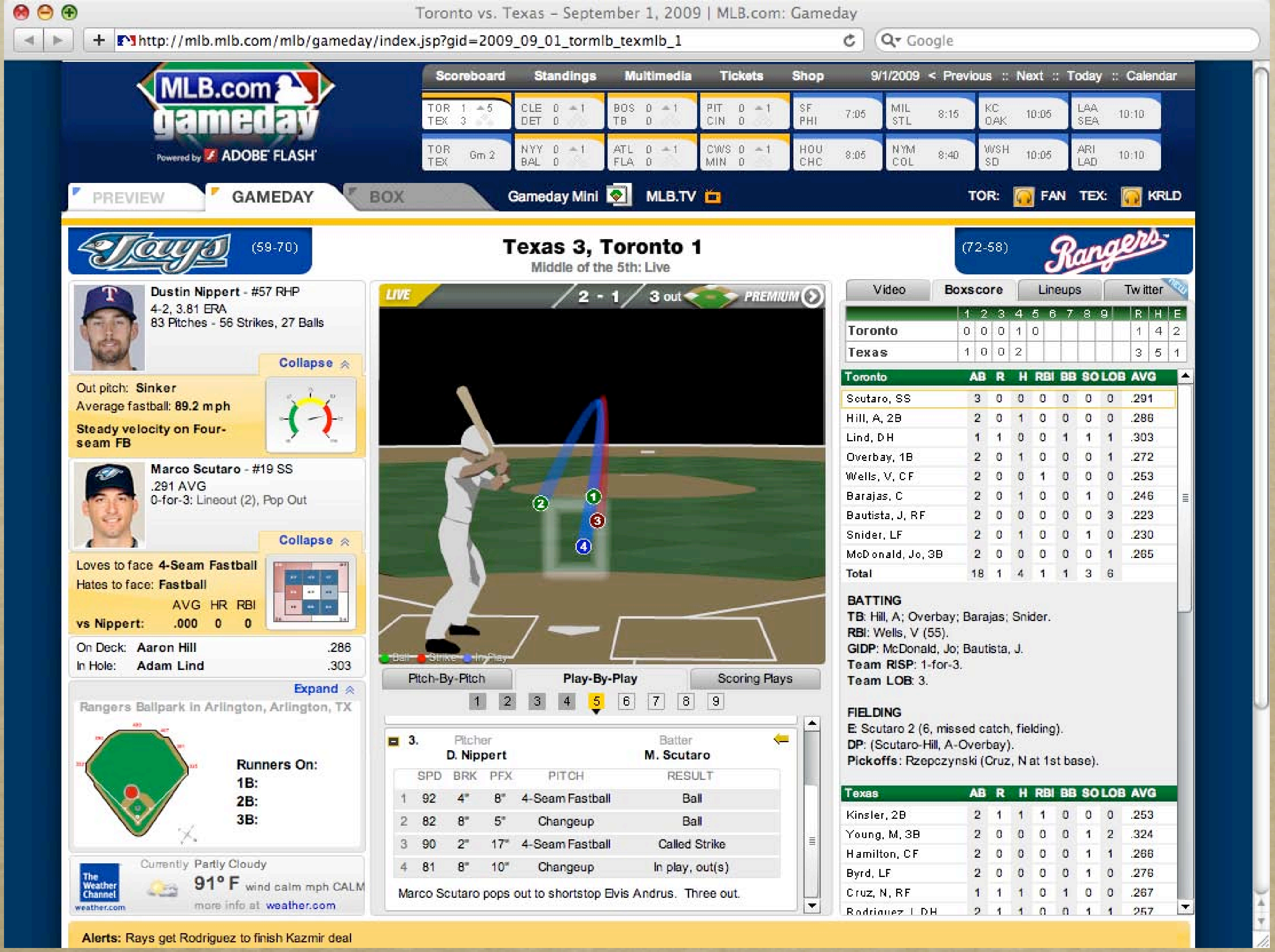
Due to implementation deficiencies, SOAP-based Web Services (and its sibling Service Oriented Architectures) are incapable of realizing the promise of fine-grained, composable services without fundamentally violating the REST axioms that permitted the web to scale.

Insight: RESTful Services

Even for hypermedia-related services - such as document management (via WebDAV) - the construction of "RESTful" services has produced inconsistent and incomplete interfaces. Non-content related services have proven even more difficult to create.

Insight: New Web Apps

In some emerging Web applications, computation has appeared as a first-class concept.



TOR 1 ▲5 / TEX 3 — CLE 0 ▲1 / DET 0 — BOS 0 ▲1 / TB 0 — PIT 0 ▲1 / CIN 0 — SF / PHI 7:05 — MIL / STL 8:15 — KC / OAK 10:05 — LAA / SEA 10:10

TOR / TEX Gm 2 — NYY 0 ▲1 / BAL 0 — ATL 0 ▲1 / FLA 0 — CWS 0 ▲1 / MIN 0 — HOU / CHC 8:05 — NYM / COL 8:40 — WSH / SD 10:05 — ARI / LAD 10:10

(59-70)

Texas 3, Toronto 1
Middle of the 5th: Live

(72-58)

Dustin Nippert - #57 RHP
4-2, 3.81 ERA
83 Pitches - 56 Strikes, 27 Balls

Out pitch: **Sinker**
Average fastball: **89.2 mph**
Steady velocity on Four-seam FB

Marco Scutaro - #19 SS
.291 AVG
0-for-3: Lineout (2), Pop Out

Loves to face **4-Seam Fastball**
Hates to face: **Fastball**

	AVG	HR	RBI
vs Nippert:	.000	0	0

On Deck:	**Aaron Hill**	.286
In Hole:	**Adam Lind**	.303

Rangers Ballpark in Arlington, Arlington, TX

Runners On:
1B:
2B:
3B:

Currently Partly Cloudy
91° F wind calm mph CALM
more info at weather.com

3. Pitcher **D. Nippert** Batter **M. Scutaro**

	SPD	BRK	PFX	PITCH	RESULT
1	92	4"	8"	4-Seam Fastball	Ball
2	82	8"	5"	Changeup	Ball
3	90	2"	17"	4-Seam Fastball	Called Strike
4	81	8"	10"	Changeup	In play, out(s)

Marco Scutaro pops out to shortstop Elvis Andrus. Three out.

	1	2	3	4	5	6	7	8	9		R	H	E
Toronto	0	0	0	1	0						1	4	2
Texas	1	0	0	2							3	5	1

Toronto	AB	R	H	RBI	BB	SO	LOB	AVG
Scutaro, SS	3	0	0	0	0	0	0	.291
Hill, A, 2B	2	0	1	0	0	0	0	.286
Lind, DH	1	1	0	0	1	1	1	.303
Overbay, 1B	2	0	1	0	0	0	1	.272
Wells, V, CF	2	0	0	1	0	0	0	.253
Barajas, C	2	0	1	0	0	1	0	.246
Bautista, J, RF	2	0	0	0	0	0	3	.223
Snider, LF	2	0	1	0	0	1	0	.230
McDonald, Jo, 3B	2	0	0	0	0	0	1	.265
Total	18	1	4	1	1	3	6	

BATTING
TB: Hill, A; Overbay; Barajas; Snider.
RBI: Wells, V (55).
GIDP: McDonald, Jo; Bautista, J.
Team RISP: 1-for-3.
Team LOB: 3.

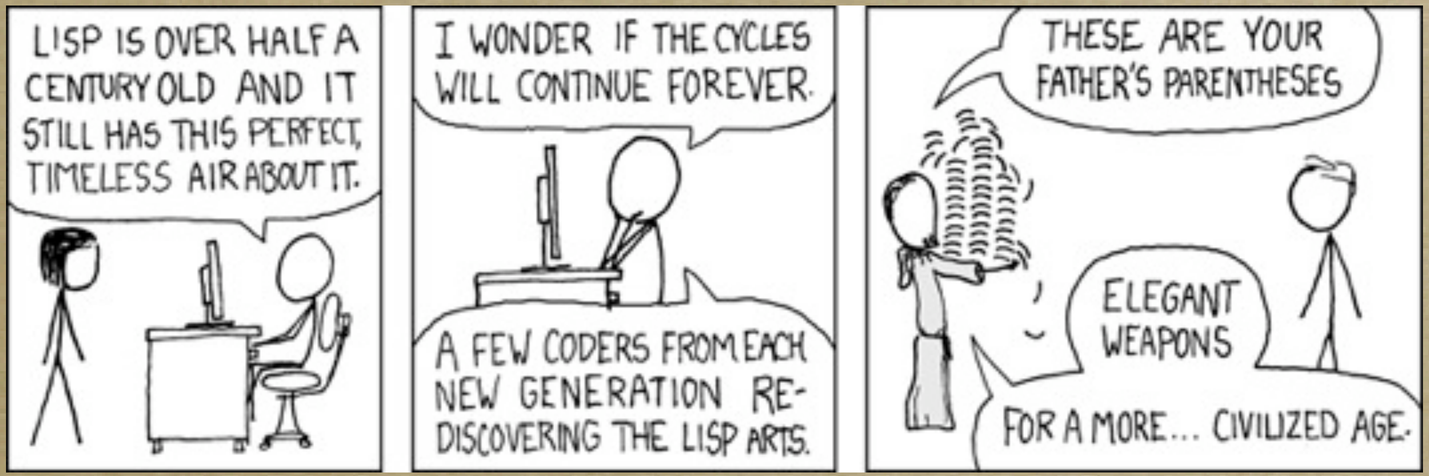
FIELDING
E: Scutaro 2 (6, missed catch, fielding).
DP: (Scutaro-Hill, A-Overbay).
Pickoffs: Rzepczynski (Cruz, N at 1st base).

Texas	AB	R	H	RBI	BB	SO	LOB	AVG
Kinsler, 2B	2	1	1	1	0	0	0	.253
Young, M, 3B	2	0	0	0	0	1	2	.324
Hamilton, CF	2	0	0	0	0	1	1	.266
Byrd, LF	2	0	0	0	0	1	0	.276
Cruz, N, RF	1	1	1	0	1	0	0	.267
Rodriguez, I, DH	2	1	1	0	0	1	1	.257

Alerts: Rays get Rodriguez to finish Kazmir deal

Mobile code

- *Mobile agents did not succeed, but code on demand and remote evaluation have found niches [Carzaniga, ICSE 2007]*
- *Remote evaluation via exchange of expressions: Scheme in Tubes [Halls, 1997]*


LISP IS OVER HALF A CENTURY OLD AND IT STILL HAS THIS PERFECT, TIMELESS AIR ABOUT IT.
I WONDER IF THE CYCLES WILL CONTINUE FOREVER.
A FEW CODERS FROM EACH NEW GENERATION RE-DISCOVERING THE LISP ARTS.
THESE ARE YOUR FATHER'S PARENTHESES
ELEGANT WEAPONS
FOR A MORE... CIVILIZED AGE.

Insight: Mobile Code

Due to the improvements in the JavaScript engines, the modern browser is far more powerful and capable today than it was in the mid-'90s. Distributed mobile code systems can be built on top of existing Web infrastructure.

Hypothesis

We can construct a set of axioms that more precisely and effectively guide the architecture of Web applications. These axioms can also further facilitate new and fundamentally different classes of applications to be deployed on the Web than the originally intended distributed hypermedia applications.

CREST Design Considerations

Computations and their expressions are explicitly **named**.

Services may be exposed through a variety of URLs which offer perspectives on the same computation; interfaces may offer complementary supervisory functionality such as debugging or management.

Functions may be added to or removed from the binding environment over **time** or their semantics may change.

Computational loci may be **stateful** (and thus permit indirect interactions between computations), but must also support **stateless** computations.

Potentially autonomous **computations** exchange and maintain state;
A rich set of stateful relationships exist among a set of distinct URLs.

The computation is **transparent** and can be inspected, routed, and cached.

The **migration** of the computation to be physically closer to the data store is supported thereby reducing the impact of network **latency**.

CREST Axioms

1. *A resource is a locus of computations, named by an URL.*
2. *The representation of a computation is an expression plus metadata to describe the expression.*
3. *All computations are context-free.*
4. *Only a few primitive operations are always available, but additional per-resource and per-computation operations are also encouraged.*
5. *The presence of intermediaries is promoted.*

CREST Design Considerations

Computations and their expressions are explicitly **named**. **(CA1, CA2)**

Services may be exposed through a variety of URLs which offer perspectives on the same computation. **(CA1)**; interfaces may offer complementary supervisory functionality such as debugging or management. **(CA4)**

Functions may be added to or removed from the binding environment over **time** or their semantics may change. **(CA4)**

Computational loci may be **stateful** (and thus permit indirect interactions between computations), but must also support **stateless** computations. **(CA3)**

Potentially autonomous **computations** exchange and maintain state **(CA2, CA3)**; A rich set of stateful relationships exist among a set of distinct URLs. **(CA1)**

The computation is **transparent** and can be inspected, routed, and cached. **(CA5)**

The **migration** of the computation to be physically closer to the data store is supported thereby reducing the impact of network **latency**. **(CA2)**

Validation

- *Characterizing the dissonance seen in existing systems against new style*
- *Creating a framework based on the new style that demonstrates novel applications*

Dissonance redux: AJAX

- *Migration and latency: Moving computation from server to client results in visually-rich low-latency applications.*
- *Mashups: Goggles, AP News feeds, etc.*
- *CREST also predicts as yet unseen forms of mashups.*

New types of mashups

- *Derived mashups: Source of a mashup is a mashup itself; combination happens on an intermediary rather than a browser*
- *Higher-order mashups: a mashup that accepts one or more mashups as input and outputs a mashup itself*

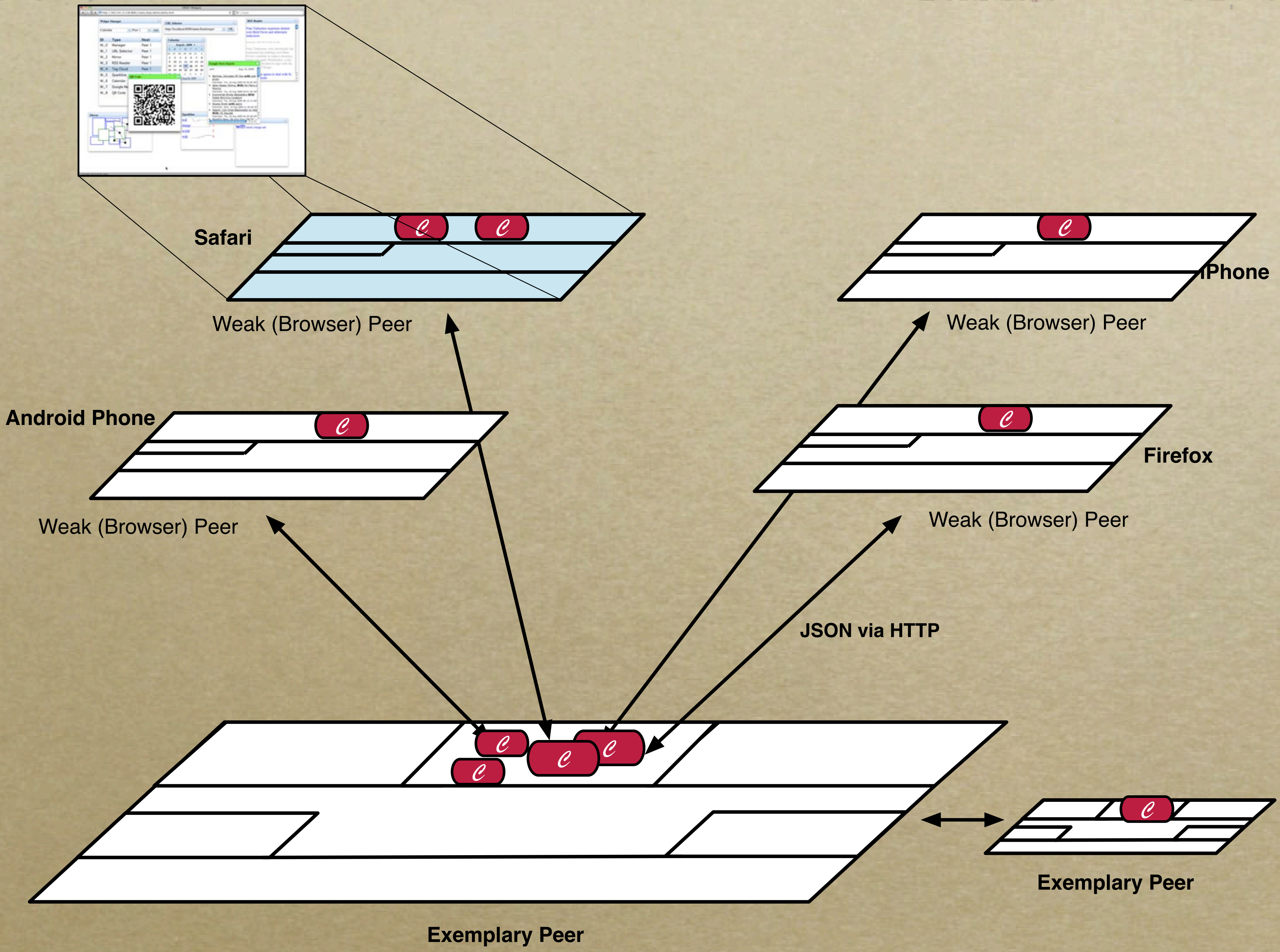
Novel applications on the Web

CREST can serve as the foundation for new classes of decentralized, Internet-based applications.

Demo

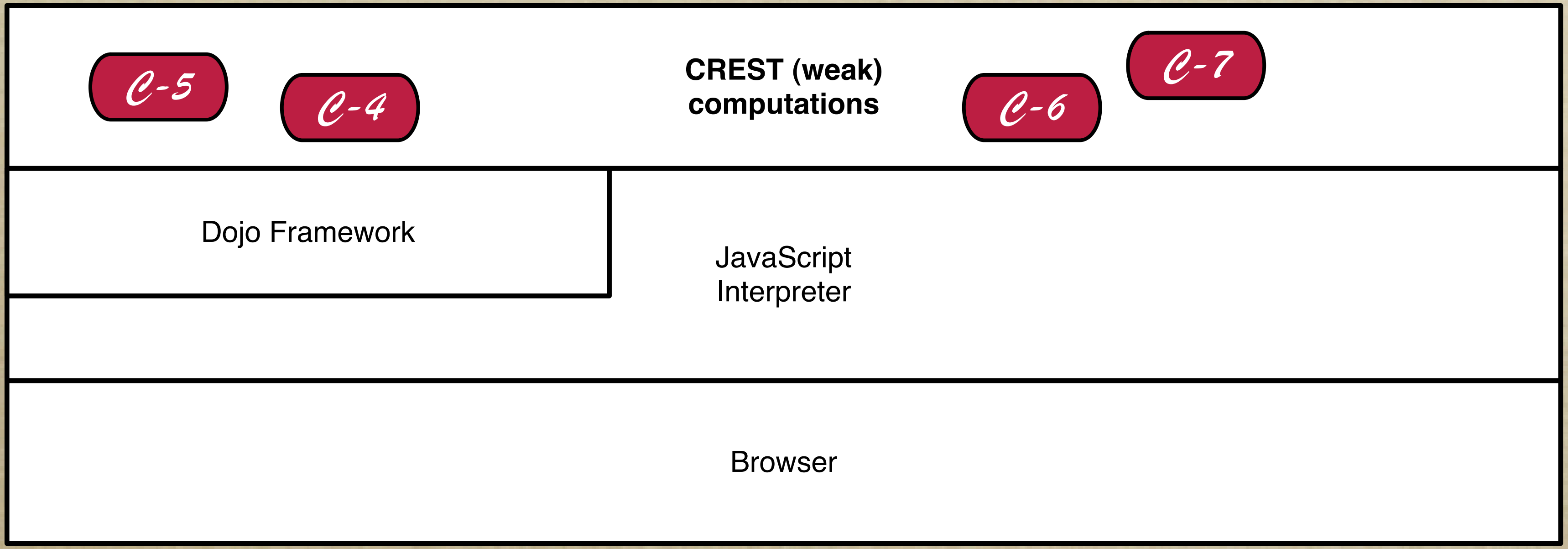
Thanks to Michael Gorlick, Yongjie Zheng, and Alegria Baquero.

CREST Overview



CREST Peers

Weak Peers



C-5

C-4

CREST (weak) computations

C-6

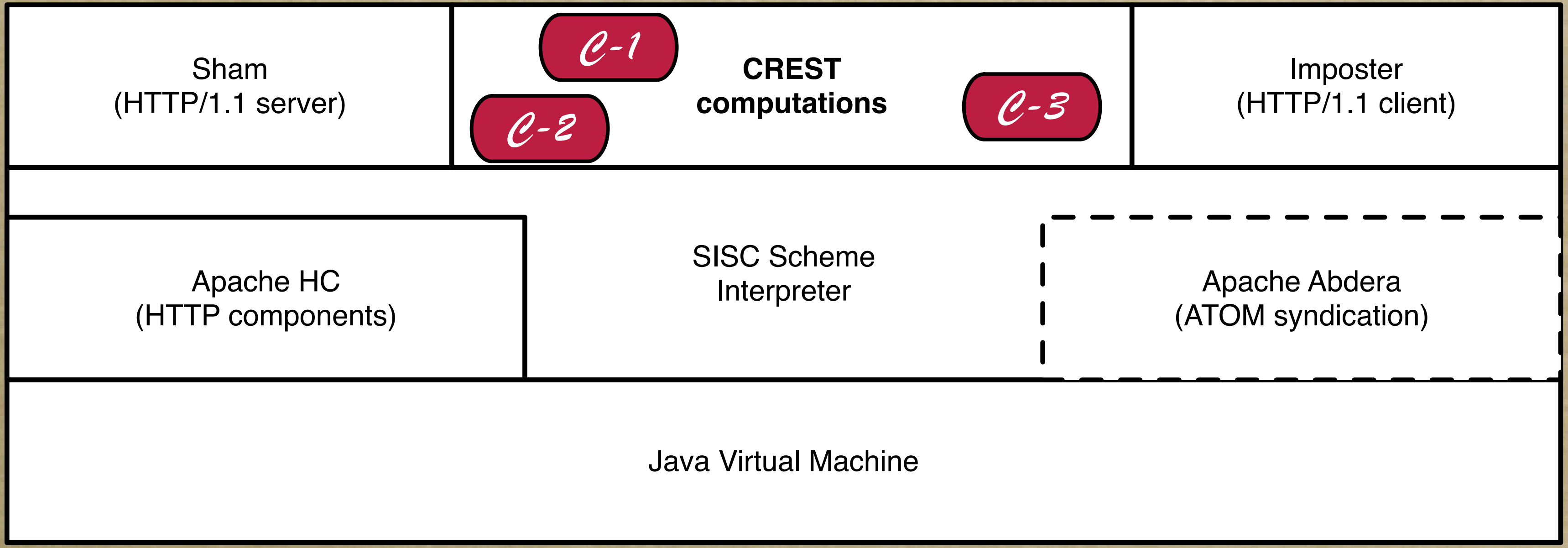
C-7

Dojo Framework

JavaScript Interpreter

Browser

Exemplary Peers



Sham (HTTP/1.1 server)

C-1

C-2

CREST computations

C-3

Imposter (HTTP/1.1 client)

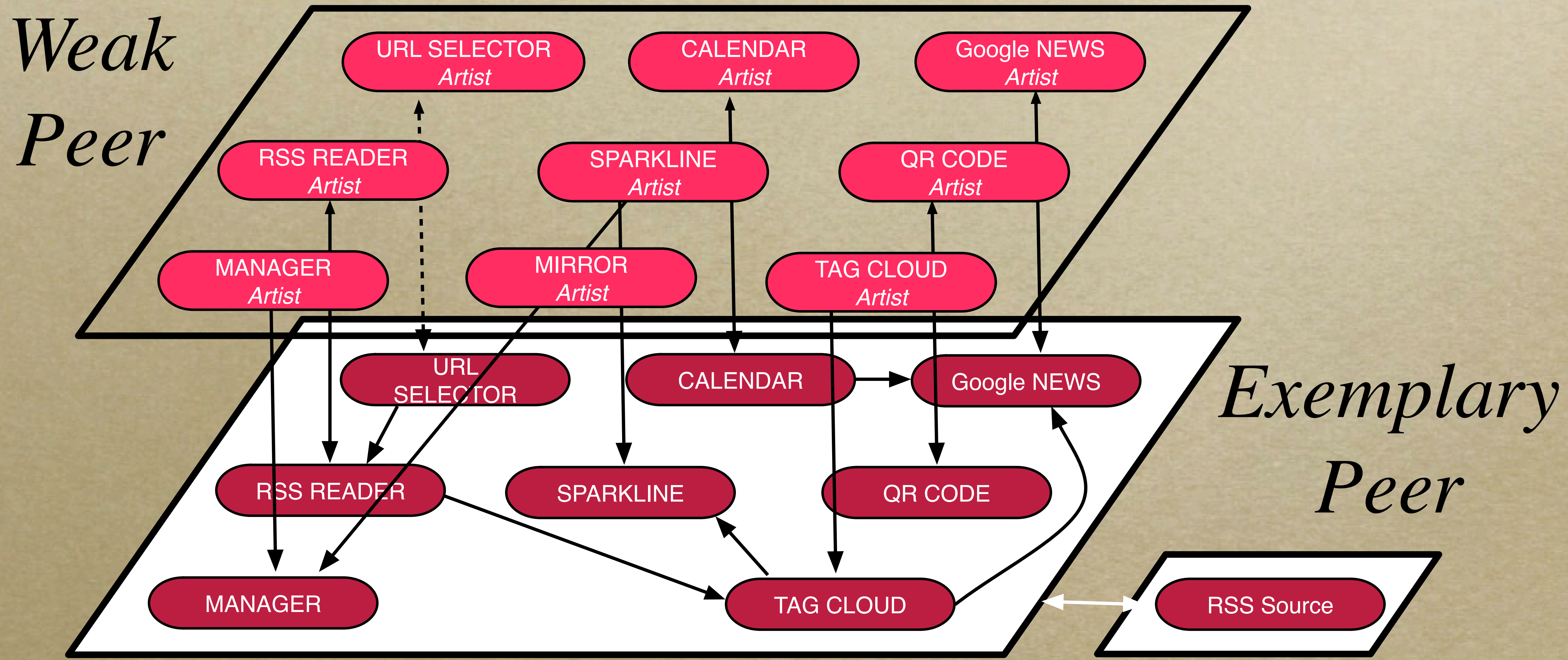
Apache HC (HTTP components)

SISC Scheme Interpreter

Apache Abdera (ATOM syndication)

Java Virtual Machine

CREST Computations



Demo (redux)

Demo FAQs

- *Isn’t this Google Wave?*
    - *Wave is just a shared XML document.*
- *Isn’t this web services?*
    - *Yes, but far more powerful than SOA.*
    - *Composability is free.*
    - *Nano-services can be installed.*

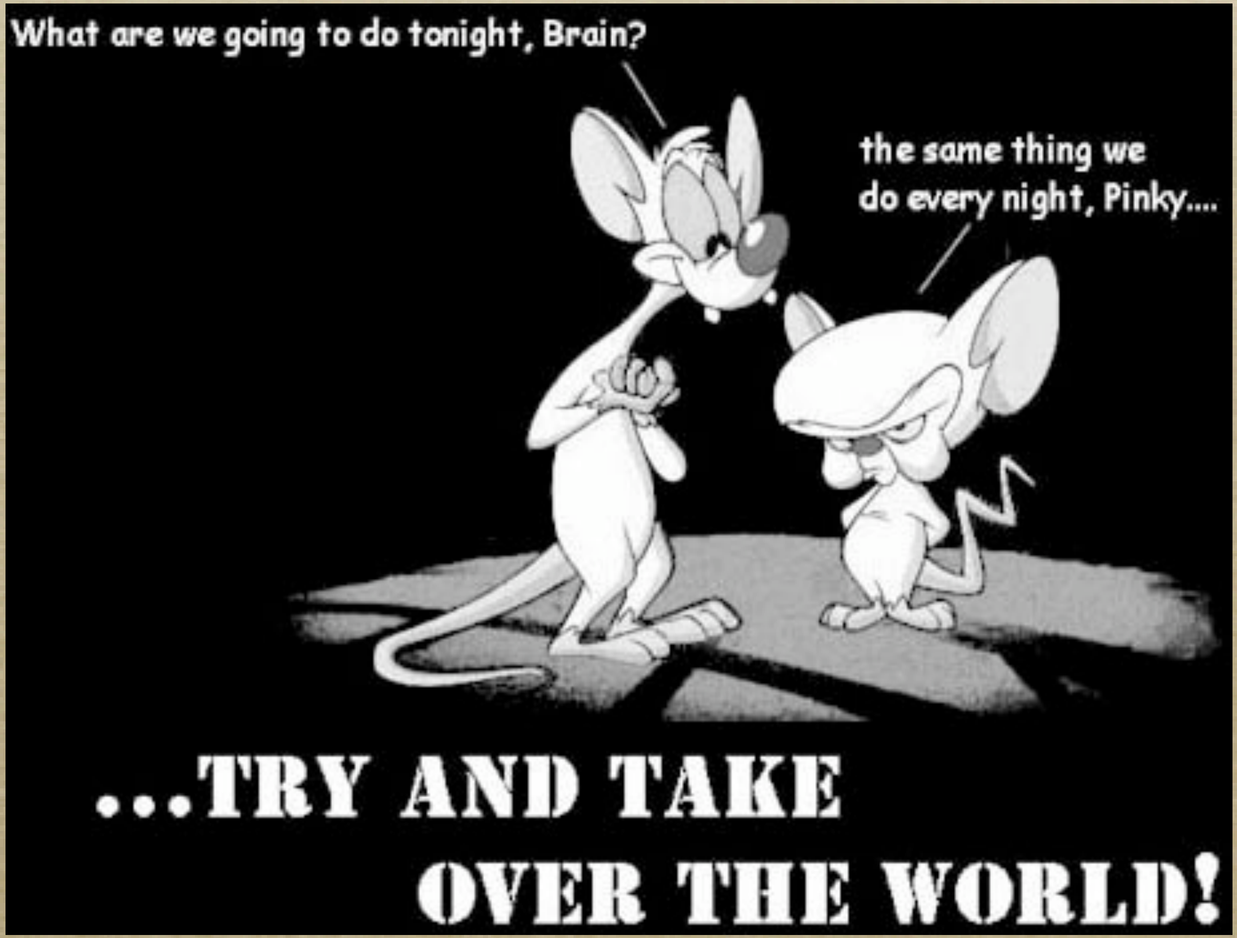
Research Question redux

What happens when dynamism is introduced into the Web?

The underlying architecture of the Web shifts, from a focus on the exchange of static content to the exchange of active computations.

Future Work

- *Gorlick: Streaming state kinematics*
- *Recombinant services*
- *Smart (power) grid, smart cargo, etc….*
- *Bring framework to Apache*


What are we going to do tonight, Brain?
the same thing we do every night, Pinky....
...TRY AND TAKE OVER THE WORLD!

Contributions

- *Analysis of the essential architectural decisions of the web, followed by generalization, opens up an entirely new space of decentralized, Internet-based applications*
- *Recasting the web as a mechanism for computational exchange instead of content exchange*
- *A new architectural style to support this recasting (CREST)*
- *Demonstrating how CREST better explains dissonance*
- *A framework for building applications backed by CREST*

Questions?