Applications for the Masses by the Masses: Why Engineers Are An Endangered Species

Todd Fast
Sun Microsystems, Inc.

TS-6537

Java

Sun microsystems

Understand how software development is already changing, away from traditional models involving mostly software engineers, to new models that involve everyone.

Identify new opportunities as software engineering experts to reach many times more users by enlisting the masses to build on your work.

Propositions

- Software engineers will be an endangered species
- High school and college students will take over your job
- And you won't mind

Agenda

- **Genetic Freaks!**
- Trends
- Applications
- The Application Spectrum
- Explosive Vampires!
- Disposable Applications
- Tools: Ever-Higher Abstraction
- Technology for the Masses
- Propositions Revisited & Summary

Software Engineers are Genetic Freaks

- Take a look around you; are these people normal?
  - Above-average intelligence
  - Above-average ability to abstract
  - Above-average interest in Monty Python and other absurdist comedies
  - Below-average tolerance for imprecise answers
  - Below-average fashion sense
  - Willing to sit in boring JavaOne™ conference session instead of out pursuing mates
- Engineers are at the edges of the population curve
- Most people are not like you and me

- The sooner you realize you're not normal, the easier this will be for you

Leading Question

- Normal people are not suited to software engineering
- But, what if relatively normal people could do a lot of what you do today?
- Let's call them *casual developers*

Casual Developers

- Not “developers” as we typically define them
  - Not self identifying as engineers
- A large majority are under the age of 30
  - Students are a big fraction
- Casually use advanced technology day-in, day-out
- Produce and consume information at dramatically higher rates
- Interact with others asynchronously and non-linearly
- Assemble personalized views of their world
- Increasingly entering the corporate workforce

Casual Developers

- 7,705 students polled, *
  - 97% own a computer
  - 94% own a cell phone
  - 76% use instant messaging.
  - 34% use websites as their primary source of news
  - 28% own a blog and 44% read blogs
  - 75% of students have a Facebook account
- Other related terms:
  - Generation Y
  - 'Net Generation
  - Web Generation
  - Google Generation
  - MySpace Generation
  - Millennials

*Source: Reynol Junco and Jeanna Mastrodicasa, Connecting to the Net.Generation: What higher education professionals need to know about today's students, NASPA; First edition (March 29, 2007)

Agenda

- Genetic Freaks!
- **Trends**
- Applications
- The Application Spectrum
- Explosive Vampires!
- Disposable Applications
- Tools: Ever-Higher Abstraction
- Technology for the Masses
- Propositions Revisited & Summary

Technology Trends

- The Java™ language is mature
- Simpler solutions are better
- Heterogeneous solutions
  - Cobble together applications from a variety of pieces
  - Services are the abstraction layer that makes this possible
- The Web is a tool, and it's everywhere
- Software delivery can't keep up with demand
- Increasing prevalence of “Web 2.0” features
  - Read-write Web
  - Participation
  - Web APIs

Social Trends

- Social computing becoming pillar of mass culture
  - Social features drive technology to the masses
- *App development* merging with *app usage* merging with *content creation* merging with *content consumption* merging with *culture*
  - Apps are a way of life
  - “The medium is the message.” - Marshall McLuhan
- Casual developers are building out the next-generation Web

A Confluence of Trends

- Widespread Internet connectivity at broadband speeds
- The ability to augment Web sites and social networks with user-defined functionality
- A constantly growing supply of interesting Web APIs
- A mass market of technically-savvy people who are
  - Eager to express themselves and contribute
  - Extremely familiar with technology
  - Steeped in the conventions of social computing
  - Feel constrained by delivery of traditional technologies
- Social networks that provide viral distribution channels
- People at the design center instead of software

Agenda

- Genetic Freaks!
- Trends
- **Applications**
- The Application Spectrum
- Explosive Vampires!
- Disposable Applications
- Tools: Ever-Higher Abstraction
- Technology for the Masses
- Propositions Revisited & Summary

Question

- How many people here build applications?

What is an Application?

An application is a piece of software that helps a user solve a specific task.

What is an Application?

- Most of us make a living writing applications for other people
  - Using powerful software tools
  - Using expertise accumulated over years
- Traditional perspective of “application”
  - Solve other people's use cases
  - Are big and take significant resources to develop
  - Live a long time
  - Only highly skilled experts can create them
- Traditional examples
  - Office productivity suites, media players, e-commerce websites, employee self-service portals

What is an Application?

- Applications come in all sizes and shapes
  - Large: SAP, Amazon.com
  - Small: UNIX shell scripts, widgets
- Really, anything can be considered an application as long as it helps a user with a task
- But, why do we need to tinker with the definition of application?
  - Because the common definition of applications is changing
  - Better to say that the scope of applications is changing

What is an Application?

- Let's talk about Web applications specifically
- Size and effort are no longer reliable criteria for judging what is an application
- Different metrics for determining value of an application
  - ~~Size and feature scope~~
  - How much usage it gets
- Applications, especially small applications, are built on platforms

What is a Platform?

A platform is a piece of software that enables applications.

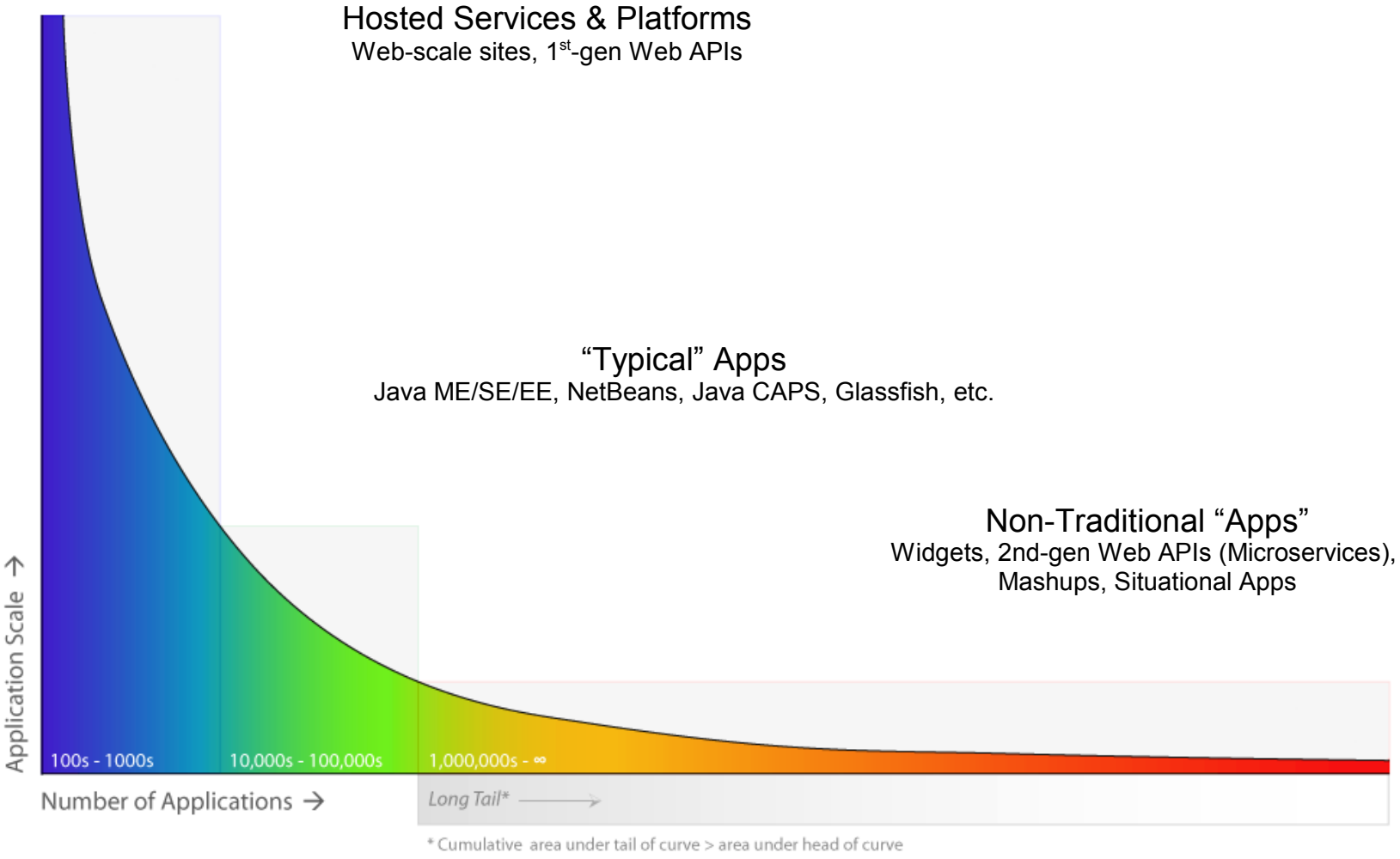
What is a Platform?

- Solve common problems so that applications don't have to
- Expose facilities (e.g. APIs) that make writing applications easier
- Are the fertilizer for ecosystems of applications built upon them
- As a rule, applications enrich the platforms they run on
  - Platforms need applications, and applications need platforms
- Platforms are more than the sum of their parts
  - Crowdsourcing enrich platforms in ways that the original developers didn't imagine, much less have time to do
  - Looks a lot like the open source model

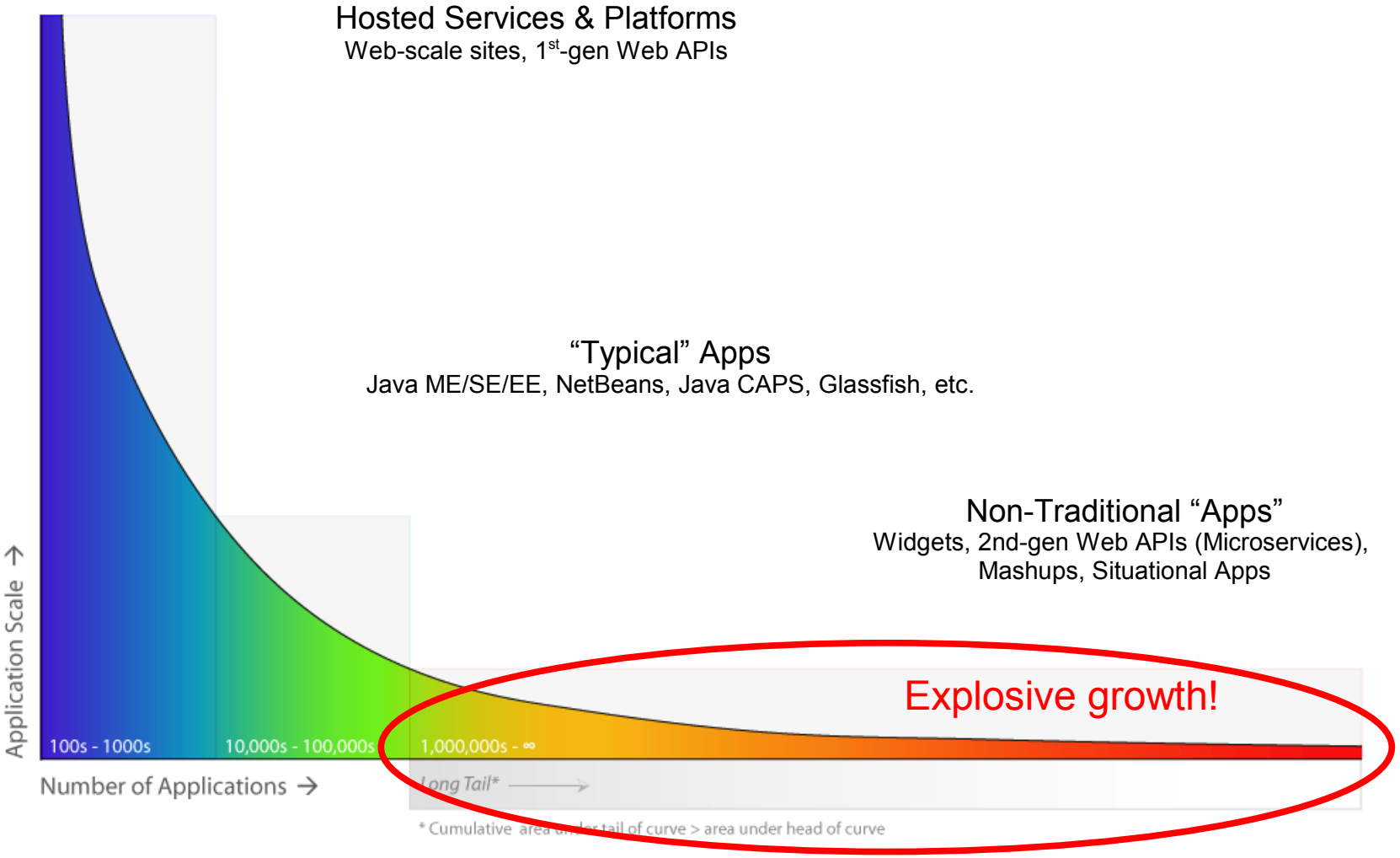
Agenda

- Genetic Freaks!
- Trends
- Applications
- **The Application Spectrum**
- Explosive Vampires!
- Disposable Applications
- Tools: Ever-Higher Abstraction
- Technology for the Masses
- Propositions Revisited & Summary

The Application Spectrum



The Application Spectrum



Kaboom!

- Explosive growth in non-traditional Web applications
  - Widgets, social apps, micro-services, mashups, situational apps
- What lit the fuse?
  - Availability of platforms that make these apps possible
    - e.g. Facebook, MySpace, Ning, Meebo
  - Mature technologies and standards for delivering these apps
  - Higher levels of abstraction in application models
  - Social computing use cases
  - Cognitive surplus
- Non-traditional apps are usually small
  - Not developed using traditional software tools or processes
  - Developed by casual developers
  - Quick to build, easy to deploy
- Where does the time come from?

Deploying Cognitive Surplus

"Where do they find the time?"

Activity	**Time Spent**
Wikipedia project	= 100,000,000 hours of human thought
Television watching in the U.S. (every year)	= 200,000,000,000 hours or **2,000** Wikipedia projects (year)

- Magnitude of available cognitive surplus is so large that even a small change could have huge ramifications

* Source: Clay Shirky (Gin, Television, and Social Surplus)

Agenda

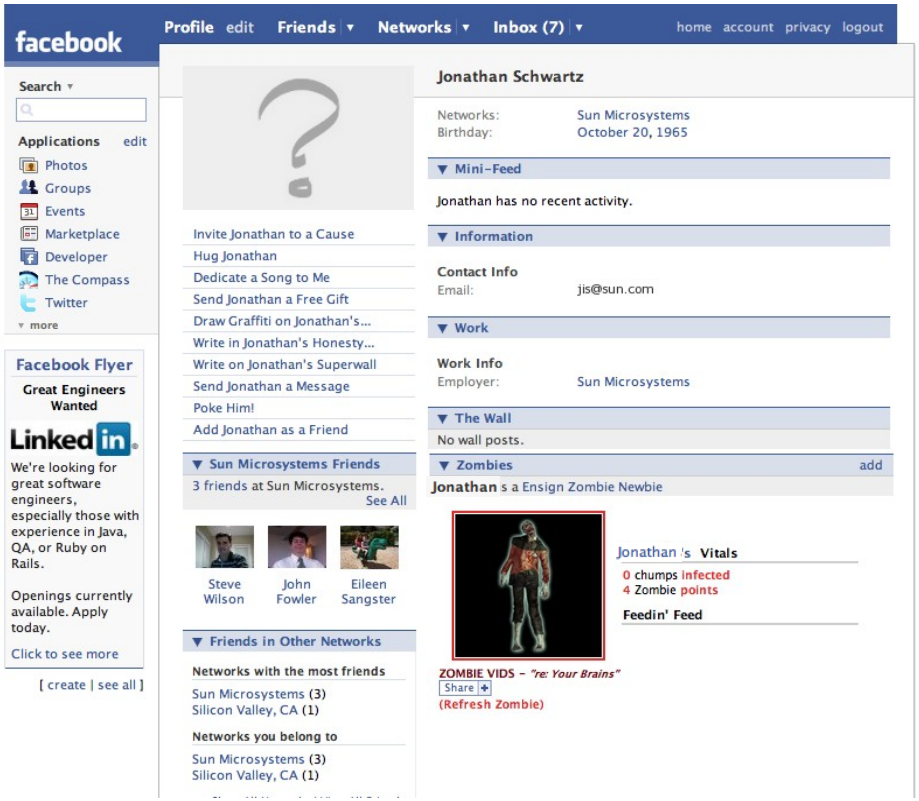
- Genetic Freaks!
- Trends
- Applications
- The Application Spectrum
- **Explosive Vampires!**
- Disposable Applications
- Tools: Ever-Higher Abstraction
- Technology for the Masses
- Propositions Revisited & Summary

Watch out!

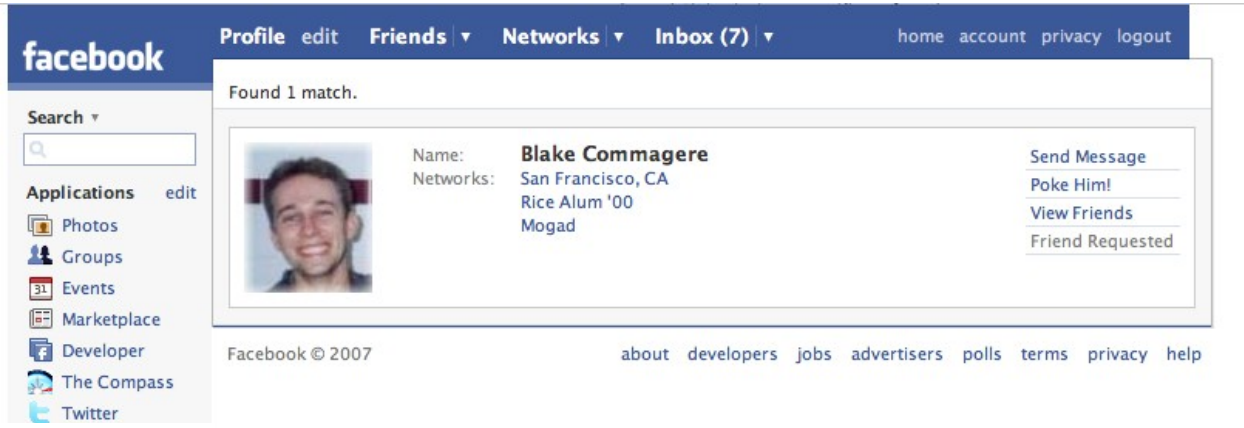
- In October 2007, ~250,000 people were turned into Zombies every day...

Watch out!

- In October 2007, ~250,000 people were turned into Zombies every day...in Facebook



...with a lone developer running the app



- Runtime system
- Database
- Servers
- Storage
- Networking
- Bandwidth
- Programming
- Security

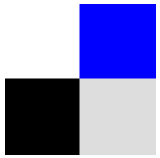
"The technical expertise and financial resources required of the (Facebook application) developer are very high." - Marc Andreessen

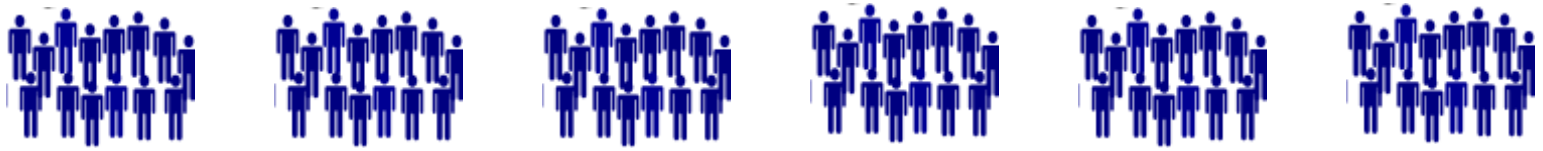
What's Facebook?

Social Operating System

Premise: Applications are more interesting when they connect people





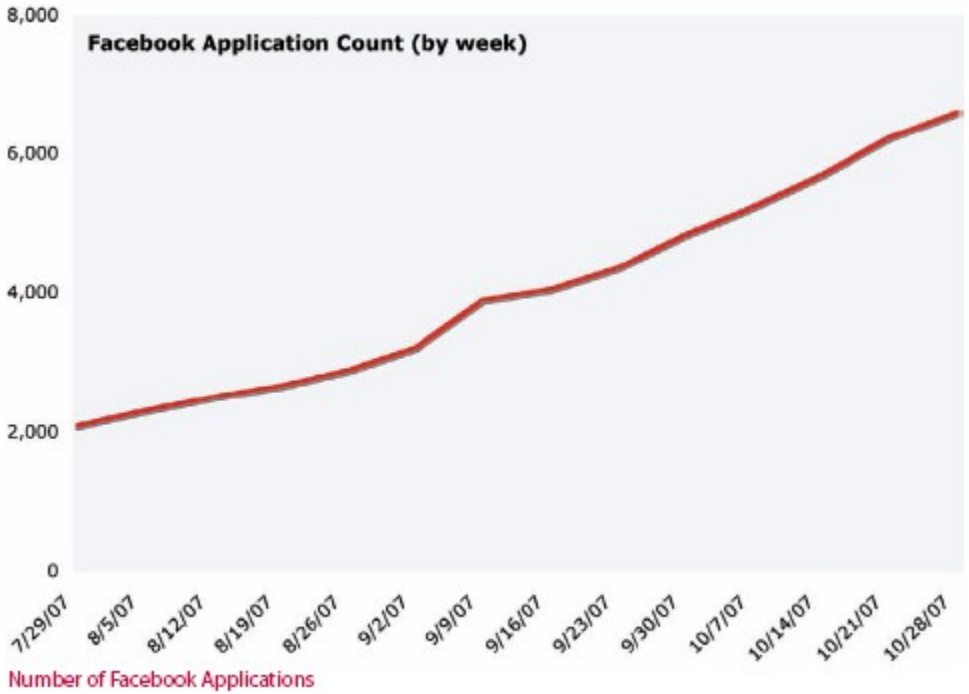



The Facebook Platform

- Mechanism for embedding 3rd party applications into the Facebook
  - Outside developers inject new features and content into the Facebook environment
- Applications are aware of who your friends are
  - Retrieve user information via callbacks to Facebook APIs
- Opportunity to build applications that can reach 70M+ users
- Highly viral distribution engine
  - Facebook news feed notifies friends when you install an application
- Example: iLike
  - May 24, 2007 — Launched music app in Facebook
  - 24 hours later — 180,000 users
  - 2 weeks later — 688,758 users
  - 3 weeks later — 2,018,442 users

Applications on Facebook

- ~2,000 to ~6,600 Facebook apps in 13 weeks*



* Source: O'Reilly Radar: The Facebook Application Platform, November 2007, 2nd edition

- About 24,000 Facebook applications today (5/7/2008)

Facebook Facts

- 24,013 applications
- 864,708,371 installs
- 200,000 developers currently evaluating the platform

The Adonomics 100™

Rank	Company	Installs	Active Users
1	Slide	95,576,286	4,999,806
2	Rock You	87,941,194	3,053,569
3	Social Gaming Network (SGN)	50,634,268	960,838
4	Zynga	41,864,617	1,995,076
5	Chainn Inc.	26,022,744	644,794
6	Blake Commagere	21,899,150	309,838
7	Flixster	18,477,034	554,311
8	Watercooler, Inc.	17,090,734	238,054
9	iLike	15,105,650	302,113
10	Jambool	13,586,080	165,771
11	TheBroth	13,313,519	118,030
12	42 Friends	13,294,217	329,707
13	SNAP Interactive	11,802,330	681,463
14	SA Ventures	11,742,734	1,409,128
15	EL Ventures	11,131,900	111,319
16	750 Industries	10,199,600	91,617
17	FrozenBear.com	8,851,523	268,149
18	J-Squared Medi	8,757,700	87,577
19	TS Ventures	8,701,850	174,037
20	GT Ventures	8,348,400	83,484

* Source: Adonomics.com (5/7/2008)

Success Kills

Startups crying out for more servers to keep up with demand

---------- Forwarded message ----------
From: "Ali Partovi" <ali@ilike-inc.com>
To: [redacted]
Date: Fri, 25 May 2007 22:50:06 -0700
Subject: URGENT - iLike needs help
Reid,

I have an urgent request. Please forward this to any Bay Area company you know that may be willing to lend or sell us spare webservers THIS WEEKEND!

Today we launched iLike on Facebook, and overnight it has maxed out our datacenter. As our CTO Nat puts it, "every time we add servers, Facebook eats the capacity we give it".

Since early last night, we've doubled its capacity, then doubled again, then doubled again, then doubled again, and we're now about to run out of servers. In its first day, our Facebook app has surpassed iLike.com in traffic and users (46,000 registered users signed up in first 20 hours, purely by viral spread). We're now shutting off features on iLike.com in order to free machines to service FB traffic. Meanwhile, our growth continues to accelerate.

unless something changes, we'll run out of servers. We desperately need dozens of new servers, THIS weekend, in the Bay Area (our datacenter is Equinix in Santa Clara)

We're emailing our board, investors, friends, asking if anybody has spare servers, or ideas on how to get them. We want front-end boxes (8-16 GB ram, 2-4 cords). Ideally a full rack of 40 machines, but really any number is fine. Offloading to Amazon S3/EC2 not an option (we've already offloaded what we could). We'll pay a significant premium to get new boxes immediately so we can sustain growth.

do you have ideas? do you know anybody that may have excess servers we could borrow for the next 5-7 days while we buy new servers of our own?

- Ali

OpenSocial

- Open web APIs for Social Networks
  - Standard “plug-in API” for social networking systems
  - Way for external social networking applications to "plug into" a common host environment (or "container")
  - Takes the Facebook platform concept and provides an open standard approach that can be used by the entire web
- OpenSocial Containers
  - Allow OpenSocial-compliant “apps” to run within it (potentially unchanged)
  - Any social networking system can be an OpenSocial “container”
    - Ning, Orkut, LinkedIn, Hi5, Friendster
- OpenSocial Apps
  - Use standard web development tools to build OpenSocial apps (HTML, JavaScript™ language)
  - Run in any OpenSocial-compliant “container”

State of the Widget Industry

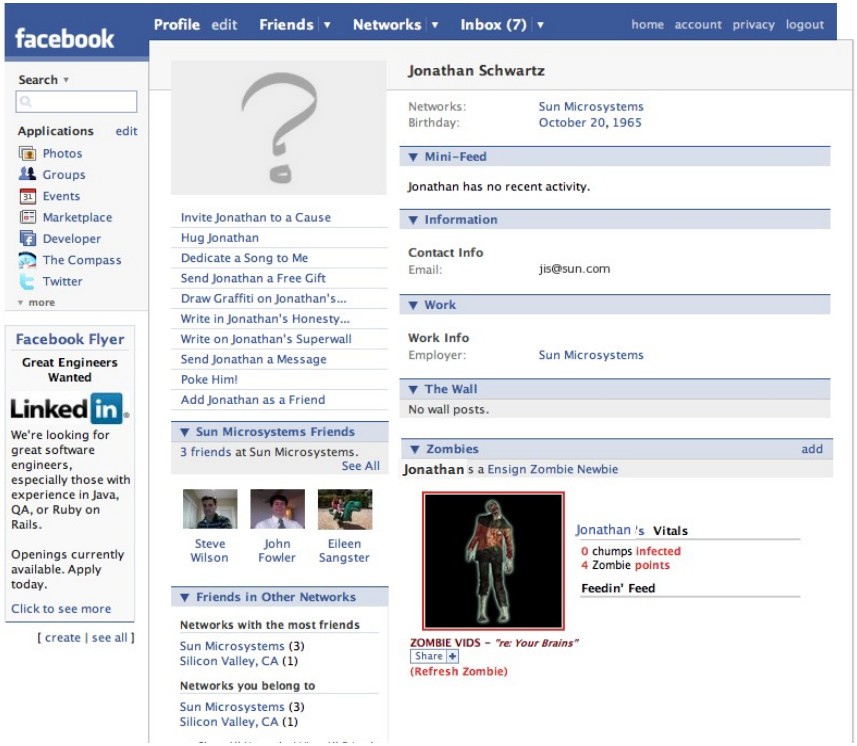
- Viewership is booming
  - Slide's slideshow widget viewed by 134M people / month
- “Social networks have created giant shopping malls for user interaction” -- Max Levchin
  - Widgets are the shop features inside malls
  - However, malls are mostly empty today
  - People are currently hanging out in the nursery (e.g. throwing sheep)
  - Movement will occur quickly (only scratching the surface of new apps)
- Monetization story is still evolving
  - Advertising? Virtual currencies? Commerce?

A Sea Change

- So if you don't believe that these small applications should be considered “real” apps...
  - How many of you have built an application that is used by 1 million people *every day*?
  - They dramatically enrich the platform on which they're running (Facebook, the Web)
- Facebook and other social platforms are major drivers of application development
- Sea change in how these applications are built
  - Not developed using traditional software tools or processes
  - Developed by casual developers
  - Usually small in scope, large in usage
- Should you feel threatened by this?
  - No. Yes. OK, maybe.

Boom and Bust

- Total daily active users for Zombies has dropped to ~67,000 (-72%)*



* Daily active user comparison from Oct, 2007 to May, 2008

Boom and Bust

- Social applications often have a limited lifespan
- Attention for social applications is essentially constant
  - Increase in one application's usage happens often at the expense of another
- If the boom and bust cycle is so short...
  - Does it make sense to think of maintaining applications long-term?
  - Does the traditional software development process fit this model?

Agenda

- Genetic Freaks!
- Trends
- Applications
- The Application Spectrum
- Explosive Vampires!
- **Disposable Applications**
- Tools: Ever-Higher Abstraction
- Technology for the Masses
- Propositions Revisited & Summary

Disposable Applications

- Cost of maintaining traditional software typically outweighs the cost to produce it
  - Assumption is that applications are long-lived
  - Non-situational, appeal to broad classes of users
- Are there classes of applications that are not long-lived?

Disposable Applications

- Situational applications (source: Wikipedia)
  - Created for small groups of users with specific needs
  - Often created and used within the group where it's used
  - Evolve to suit changing requirements
  - Short life span
  - Thrown away when requirements change too much
  - Examples:
    - UNIX shell scripts, wikis, departmental apps, 4GL environments
- Social applications
  - Usually created by individuals or small groups
  - Spread quickly through viral channels
  - Usage withers unless genuinely useful on a regular basis
  - Thrown away when something new/better/interesting comes along
  - Examples:
    - Widgets, Facebook applications, mobile applications

Disposability

- Traditional software development has a fixed cost basis
  - Below this threshold, it's not worth doing
- Disposable applications fill a niche
  - Where use cases are too small to attract resources
  - Where software development expertise is limited
  - Where requirements change frequently
- To address the long tail of use cases, we need disposable applications
  - Quality can be lower because lifespan is limited
  - Can grow into non-disposable apps if prove enduringly useful

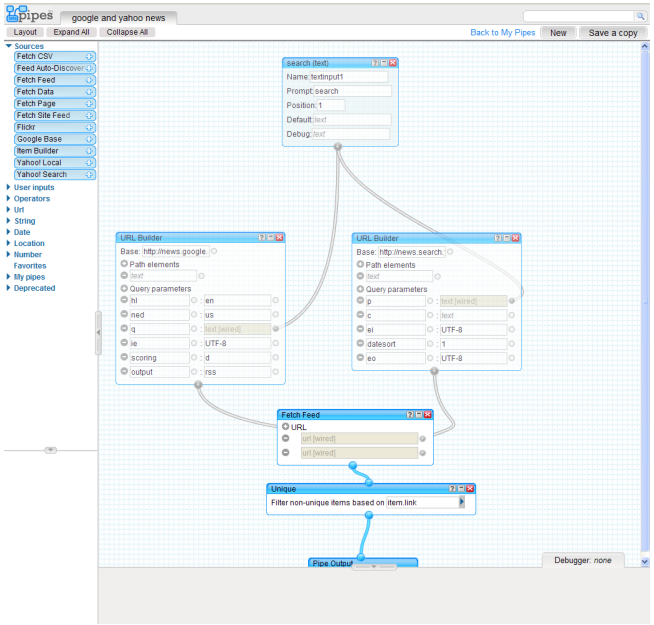
Agenda

- Genetic Freaks!
- Trends
- Applications
- The Application Spectrum
- Explosive Vampires!
- Disposable Applications
- **Tools: Ever-Higher Abstraction**
- Technology for the Masses
- Propositions Revisited & Summary

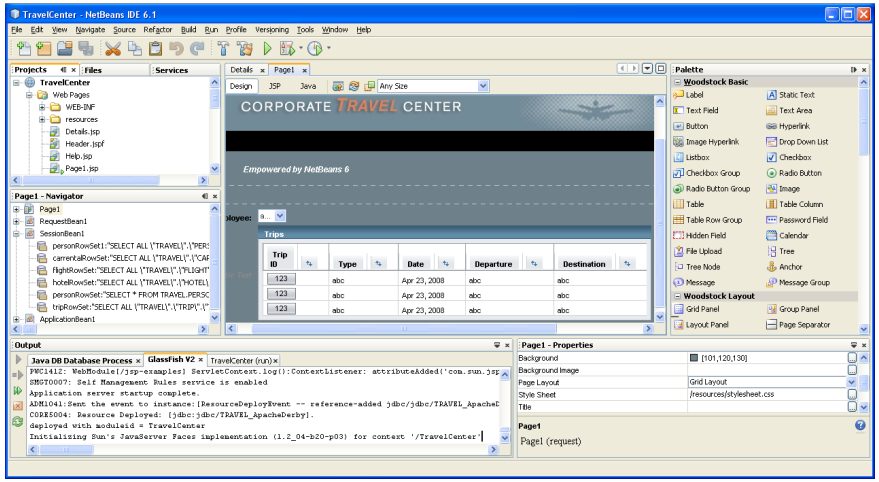
The Tools Cycle

- Tools are integral to any kind of application development
- Software tools evolve alongside the problems they solve
  - Tools are applications, too
- The purpose of software tools is twofold
  - To make things easier (usually by increasing abstraction)
  - To increase productivity of the user
- Tools don't have to be complicated
  - MediaWiki (Wikipedia) is a tool for mass collaboration
- Quantum leaps in abstraction are accompanied by new kinds of tools
- Human generations matter
  - Each generation brings with it their
    - Assumptions of how things should work
    - Ignorance of how things have worked

Abstraction vs. Capability



vs.



Todd's Tools Rules

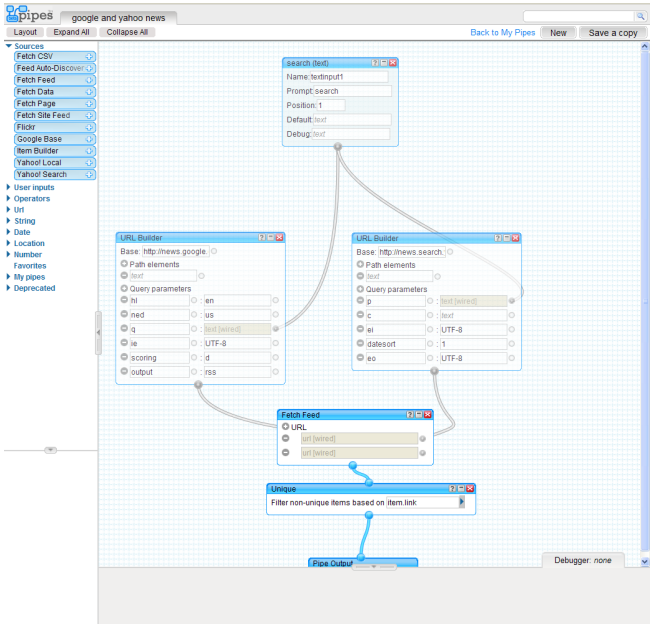
- Three factors govern a user's ability to solve a problem using a tool:

$$\text{“doability”} = \frac{\text{(amount tool does for user) (ability to address problem)}}{\text{(difficulty of solving problem)}}$$

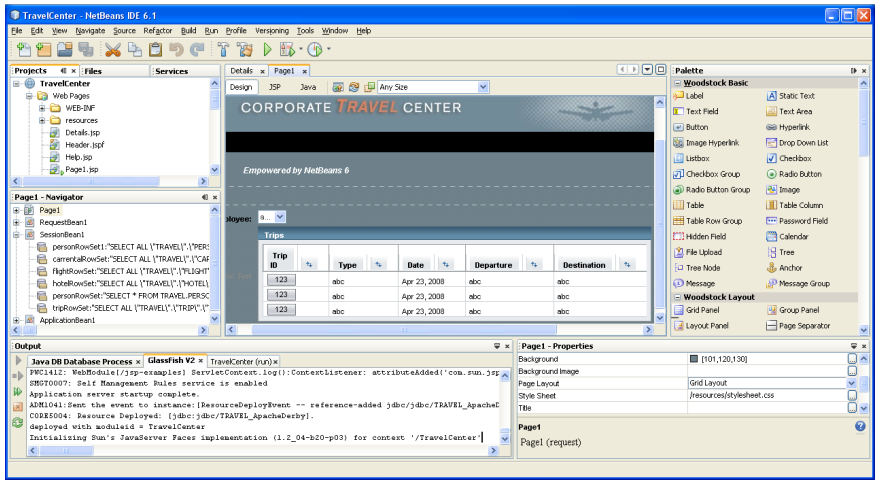
- Increasing the “doability” of a problem lets more people tackle that problem
- The amount that a tool abstracts away details is inversely proportional to the set of problems it can solve

(level of abstraction) (set of solvable problems) = 1

Abstraction vs. Capability



vs.

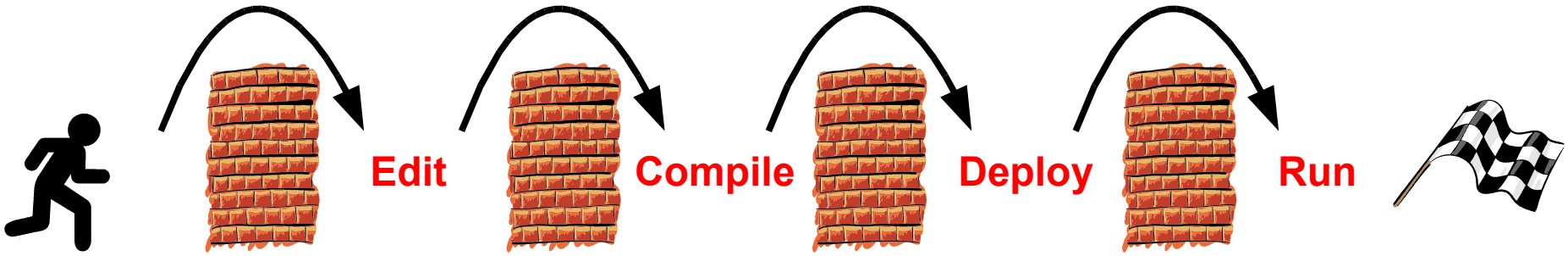


More doable, less capable

Less doable, more capable

Development 1.0

- Lots of barriers to build an application
- Software developers are comfortable with this process
  - Takes years to learn



Development 2.0

- Ever-higher levels of abstraction
- New application models
  - As abstraction goes higher, tools and application models change
  - Innovation through combination
  - Loosely coupled systems (REST-based web services)
- Web is the target platform
  - Software development delivered as a service
  - Applications delivered as services
- Participatory development
  - Barrier to participation is near zero
  - Harnesses collective intelligence of entire developer ecosystem
  - The more people use “it”, the better “it” is
    - Applies to both applications and platforms
  - Focus on peer production
  - Harnessing richness of the long tail

Tools Takeaway

- As we see new kinds of tools that raise the level of abstraction, we see increasingly greater numbers of people able to create applications
- As tools for casual developers proliferate, we see applications built by casual developers proliferate
- Cognitive surplus can (and will) be used to build applications when “doability” reaches a critical threshold

Agenda

- Genetic Freaks!
- Trends
- Applications
- The Application Spectrum
- Explosive Vampires!
- Disposable Applications
- Tools: Ever-Higher Abstraction
- **Technology for the Masses**
- Propositions Revisited & Summary

High Technology

- High technology == cutting edge technology
  - Produces the platforms and tools that lets others, with less expertise, produce something of value
- 10,000 hours to become an expert in just about anything
  - About 5 years of full-time employment
- Today, software development requires most people to be high technology experts

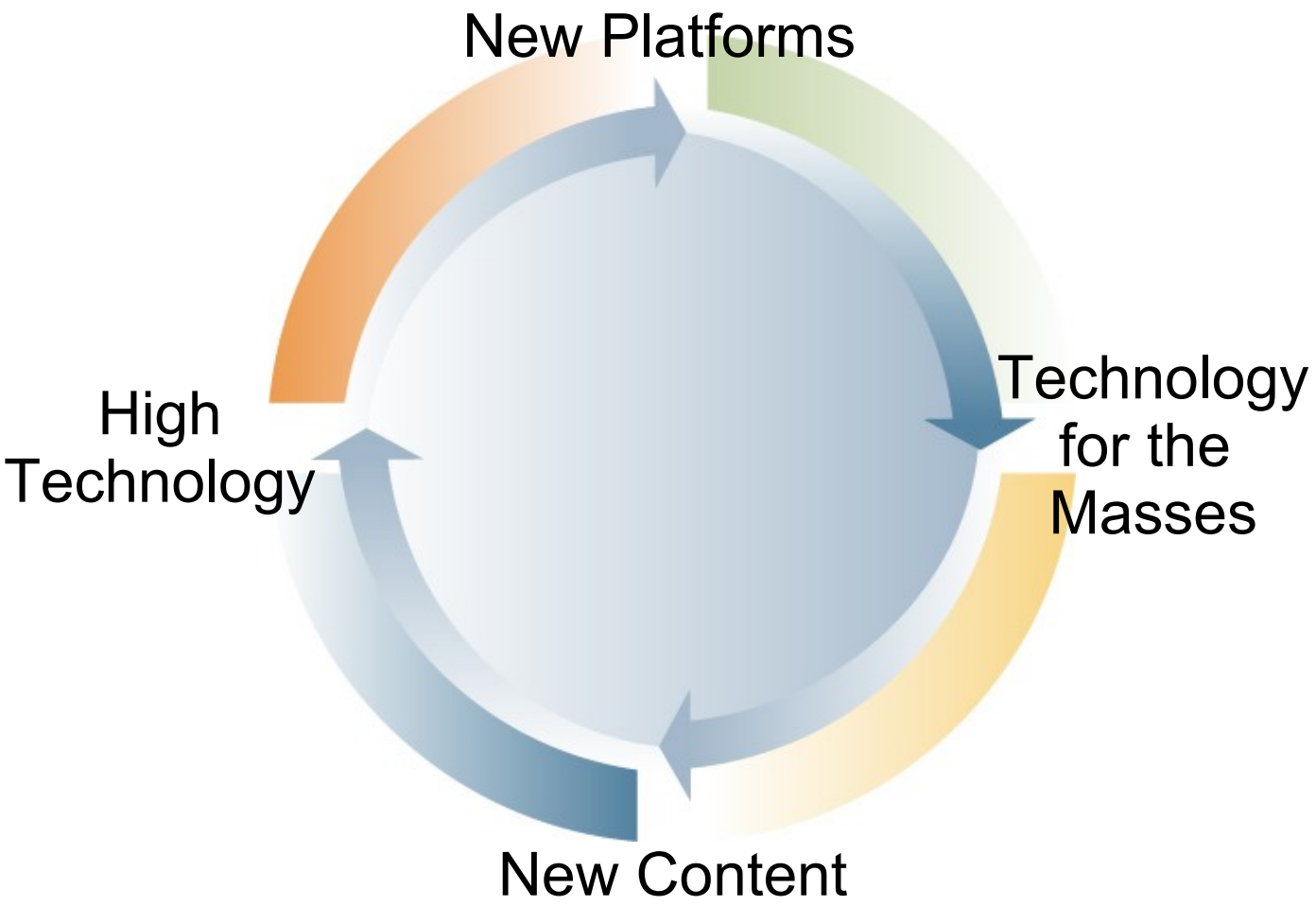
Technology for the Masses

- Focuses on use cases that are uninteresting or unfeasible for traditional engineering
- FUBU = For us, by us
- Typically starts with small but interesting use cases
  - Grows if enduringly useful
- Minimum of resources required to get started
- An individual or small group of individuals can do it
- Complete tasks can be comprehended by individuals
  - Avoids increasing specialization in normal engineering practice

Contrasting Technology Types

- Complement each other to fulfill both the head and the long tail of application use cases
- Benefits of high technology
  - Expands the scope of what can be done
  - Fulfills the head of application spectrum use cases
  - Enables communities of content creators to build out the long tail
  - By enabling others, democratizes the use and reach of technology
- Benefits of technology for the masses
  - Fulfills the long tail of application spectrum use cases
  - Drives use of platforms at an exponential rate
  - Creates massive opportunities for monetization
  - Involves increasing numbers of people in the Web collective
  - Means of democratic involvement in technology

Technology Ecosystem



Agenda

- Genetic Freaks!
- Trends
- Applications
- The Application Spectrum
- Explosive Vampires!
- Disposable Applications
- Tools: Ever-Higher Abstraction
- Technology for the Masses
- **Propositions Revisited & Summary**

Propositions Revisited

- Software engineers will be an endangered species
  - We already are, and always will be

Propositions Revisited

- Software engineers will be an endangered species
  - We already are, and always will be
- High school and college students will take over your job
  - You won't be needed to do what you're doing now: building everyday applications

Propositions Revisited

- Software engineers will be an endangered species
  - We already are, and always will be
- High school and college students will take over your job
  - You won't be needed to do what you're doing now: building everyday applications
- And you won't mind
  - Because rather than building the applications themselves, you'll be building platforms (meta-applications) so that other people can build them

Summary

- Application development model is changing
  - Opening up to a much broader audience
- Exponentially more apps will be built
  - Extending the long tail
  - Using technologies that raise the level of abstraction so that novices can accomplish useful tasks
  - Not because there are exponentially more engineers
    - Because engineers are building platforms to exponentially enable others
  - Traditional software development processes will not be able to keep up with application demand
- Software engineers will increasingly build platforms, not applications
  - The increasing pull for applications will create new economics that drive engineers to work lower in the stack

THANK YOU

Todd Fast

TS-6537