



IBM Research Division

# The Future of End User Programming?

## ICSE 2008

Sam S. Adams  
IBM Distinguished Engineer  
IBM Research



# BEGIN

- **Computers were designed and built to solve user problems**
  - ❖ **Weaving**
  - ❖ **Ballistics**
  - ❖ **Code making/breaking**
  - ❖ **Tabulating and Sorting data (cards)**
- **Programming languages were designed for end users**
  - ❖ **FORTRAN - Scientists and Engineers**
  - ❖ **COBOL - Accountants**
  - ❖ **LISP, ALGOL, APL - Mathematicians**
- **To use the new power of computers, you needed to be able to program**



# The rise of the cult of computing

- **Initially “programmers” focused on solving their own non-programming problems using computers**
- **The business success of computing created a large demand for computers and programmers**
- **Large numbers of people were trained to be general-purpose programmers**
- **Programming and later Software Engineering became a discipline and community of its own...**



## And the acolytes of ALGOL began meditating...

- **Isolated from their users in raised floor IT temples and ivy-covered academic monasteries, the navel-gazing began**



# Computing became an end in itself

- ❖ **True Believers took sides and the Wars began**
- ❖ **The Language War (aka, the 100 years War)**
  - 1000s of programming languages (<5% user oriented)
  - Currently in Détente, or at least M.A.D (Mutually Assured Distribution)
- ❖ **The Database War**
  - Truce declared (mostly) - SQL
- ❖ **The Methodology Wars I & II (pre/post Objects)**
- ❖ **The (Open) Source Wars**
- ❖ **The Browser War**
- ❖ **Etc.**



# WHILE (sweng.navel\_gazing)

- **Compelled to learn what they could of computing to solve their own problems, end users continued to adapt computing and programming to suit their own needs**
  - ❖ **Mainframe**
    - Multiple Terminal desktops and retyping
    - BASIC and REXX
  - ❖ **Textual PCs**
    - Spreadsheets
    - BASIC
    - dBase
  - ❖ **GUI PCs**
    - HyperCard
    - 4GLs (Access, et.al.)
  
- **A few drank from the fountain of Computer Science and got hooked, but most gargled their way to relatively primitive but good enough solutions....because....**



# HERESY!

- **THEY DIDN'T REALLY CARE ABOUT COMPUTERS**
  - ❖ **OR PROGRAMMING**
  - ❖ **OR ALL THE MARVELOUS COMPLEXITY!**

**<gasp>**

- **They just wanted the computer to help them with what they really cared about!**



# **#if ICSE\_Attendee define End\_User\_Programmer**

- **So what is an End User Programmer?**
  - ❖ **Someone who programs computers for their own use**
  - ❖ **Someone more interested in the running program than the process of making it**
  - ❖ **Someone who doesn't wake up in the morning thinking how cool programming is!**
  
- **Common assumptions about EUPs**
  - ❖ **Baby programmers**
    - **Everything needs to be predigested**
    - **Can't handle complexity, abstraction, specification....**
    - **Can't be trusted with powerful and dangerous tools**
  - ❖ **Need to be fixed to become REAL Programmers**



# A counter example

Binary "Machine Code"

Assembler

```
slip 1  
p 13  
p 2 tog  
p 1  
turn  
slip 1  
k 6  
k2 tog left  
k 1  
turn  
slip 1  
p 7
```



# Grandmothers as Real Programmers?

## American Flag Socks

**One adult size fits all - leg and foot length can be adjusted as needed.**

**Materials:** One skein each (220-yd) red, white and blue.

One set each size 4 and 6 (3.5 and 4 mm) double-pointed needles, or size to obtain correct gauge.

**Gauge:** In Stockinette stitch, 5 sts = 1"

**Cuff:** With larger needles and blue, cast on 52 sts loosely. Divide sts evenly on smaller needles and place a yarn marker at the beginning of round. Work around in k1, k1 rib for 1".

**Leg:** Change to larger needles, then knit around with blue, inc 2 sts in round - 54 sts. Divide sts by putting 18 sts on each needle.

Continue knitting every round following Star Chart from round 1 to 24. Next round, knit with red dec 3 sts evenly around-51 sts remain.

Knit 3 more rounds red. Next round, knit with white dec 2 sts evenly around-49 sts remain. Work 3 more rounds white. Work for 4 more rounds red, and 4 more rounds white (stripe pattern). Divide sts as follows for heel: place first 12 sts and last 11 sts on one needle for heel, place remaining 26 sts on holder for instep.

**Heel:** Working on these 23 sts only, with red, work as follows:

**Row 1 (wrong side):** Slip first st, purl across.

**Row 2:** Slip first st, knit across.

Repeat these 2 rows until you have worked 18 rows. Place marker on this row.

**Turn heel as follows:** With wrong side facing, slip 1, p 13, p 2 tog, p 1, turn work; slip 1, k 6, k2 tog left, k 1, turn; slip 1, p 7, p 2 tog, p 1, turn; slip 1, k 8 sts, k2 tog left, k 1, turn; slip 1, p 9, p 2 tog, p 1, turn. Continue in this manner, working toward sides of heel and having 1 st more between the decreases on each row until 15 sts remain.

**Gusset and Foot:** With right side facing and red, start at the center of the 15 heel sts and knit across last 7 sts from heel, pick up and knit 10 sts on side of heel, work across 26 sts from instep holder, pick up and k 10 sts on side of heel, and work across 8 remaining sts from heel. Place yarn marker at beginning of row - you should have 17 sts on first needle, 26 sts on second needle, and 18 sts on third needle (61 sts total). Knit 1 round red.

**Decrease Round:** Knit with red to within last 3 sts on first needle, k2 tog, k 1; k across sts on second needle; on third needle, k 1, k2 tog left, k to end of row. While keeping stripe pattern as established, repeat these last 2 rounds 5 more times - 49 sts remain. Work even until foot length is 7" to 9" from heel marker, as desired. Place 12 sts on first and third needles, 25 sts on second needle.

**Toe:** Work with next color (either red or white).

**Round 1:** Knit to within 3 sts from end of first needle, k2 tog, k 1; on second needle, k 1, k2 tog left, knit to last 3 sts, k 2 tog, k 1; on third needle, k 1, k2 tog left, knit to end.

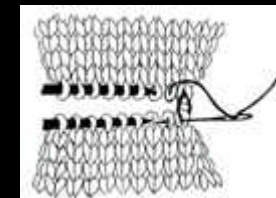
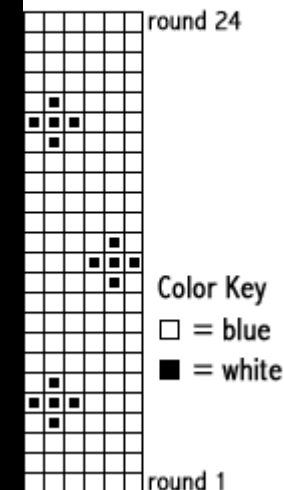
**Round 2:** Knit.

Repeat these 2 rounds 5 more times, then repeat Round 1 every round until 13 sts remain.

**Close toe using Kitchener stitch (also known as grafting or weaving - see how-to at left).**



## STAR CHART





# End User Progs Compared to Professionals

- **Able to work with formal notations (especially if related to domain of interest)**
- **Prefer concrete and tangible to abstract and general**
- **Less tolerant when assumptions are violated**
- **Not helped by things like simplistic metaphors**
- **Not helped by visual programming, at least not visual spaghetti**
- **Still need to test, debug, and manage change**
- **Could benefit from many of the same sorts of tools as professionals (e.g., source debuggers, program visualization) but need them in situ, not in a large general purpose tool box**
- **NEEDED: End User Software Engineering!**



# Time.is\_now?

- **Why should the ICSE community care more about End User Programmers today than they have in the past?**
- **Because the software world is changing in big ways**
- **Much of your relevancy depends on it**
- **Its the largest opportunity you have to change the future for the better**



# Megatrends

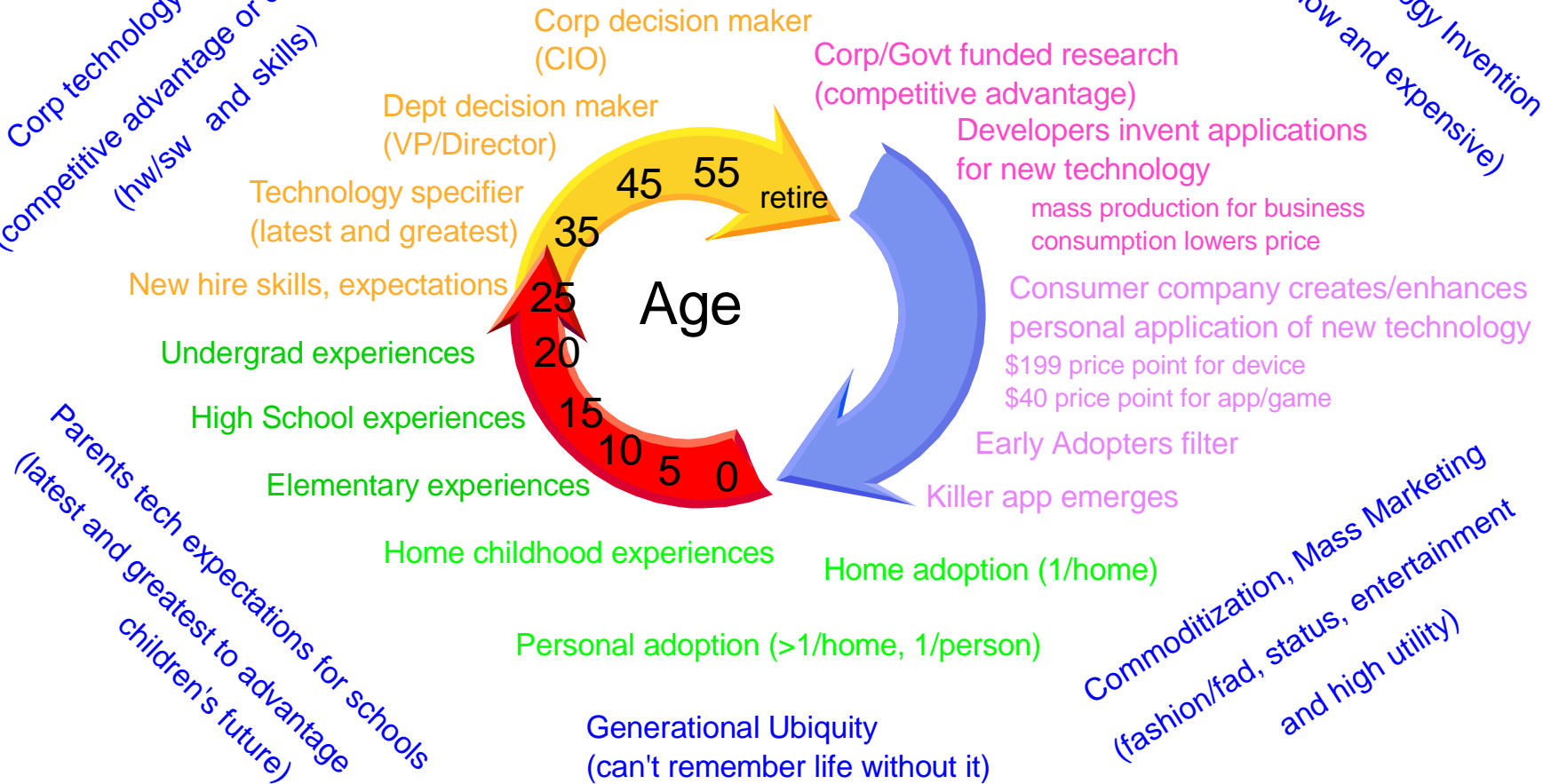
- ❖ **Generational change and expectations**
- ❖ **The evolution of abstraction and infrastructure**
- ❖ **Web 2.0 and the real Open Source Movement**
- ❖ **The Really, Really Long Tail of Software**
- ❖ **Technology evolution and End User Programming waves**
- ❖ **The coming massively multicore software (r)evolution**



# Today's Playstation kids are tomorrow's CIOs

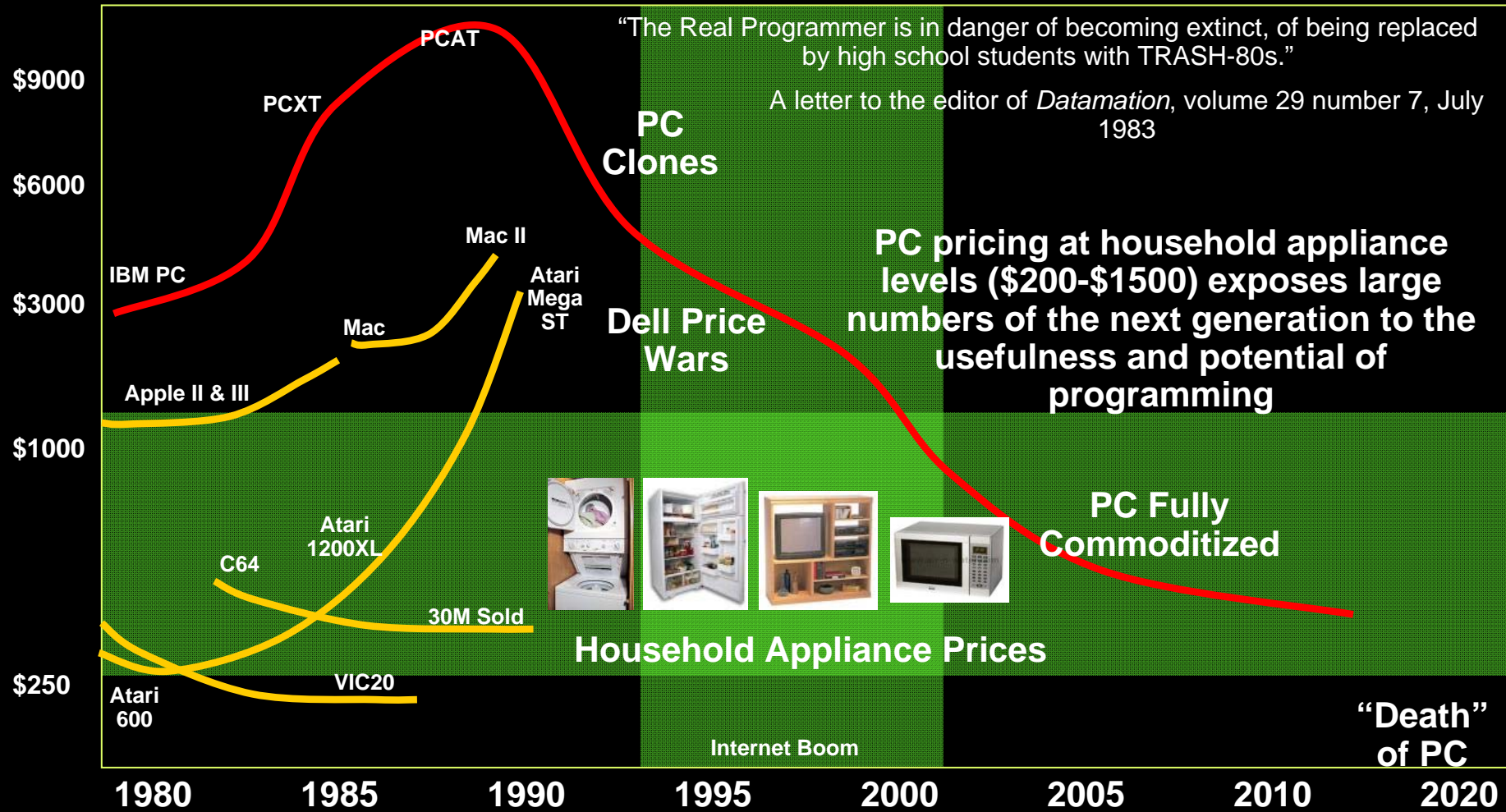
Corp technology adoption  
(competitive advantage or defense)  
(hw/sw and skills)

Technology Invention  
(slow and expensive)





# Evolution of Programmers – PC Pricing Trends

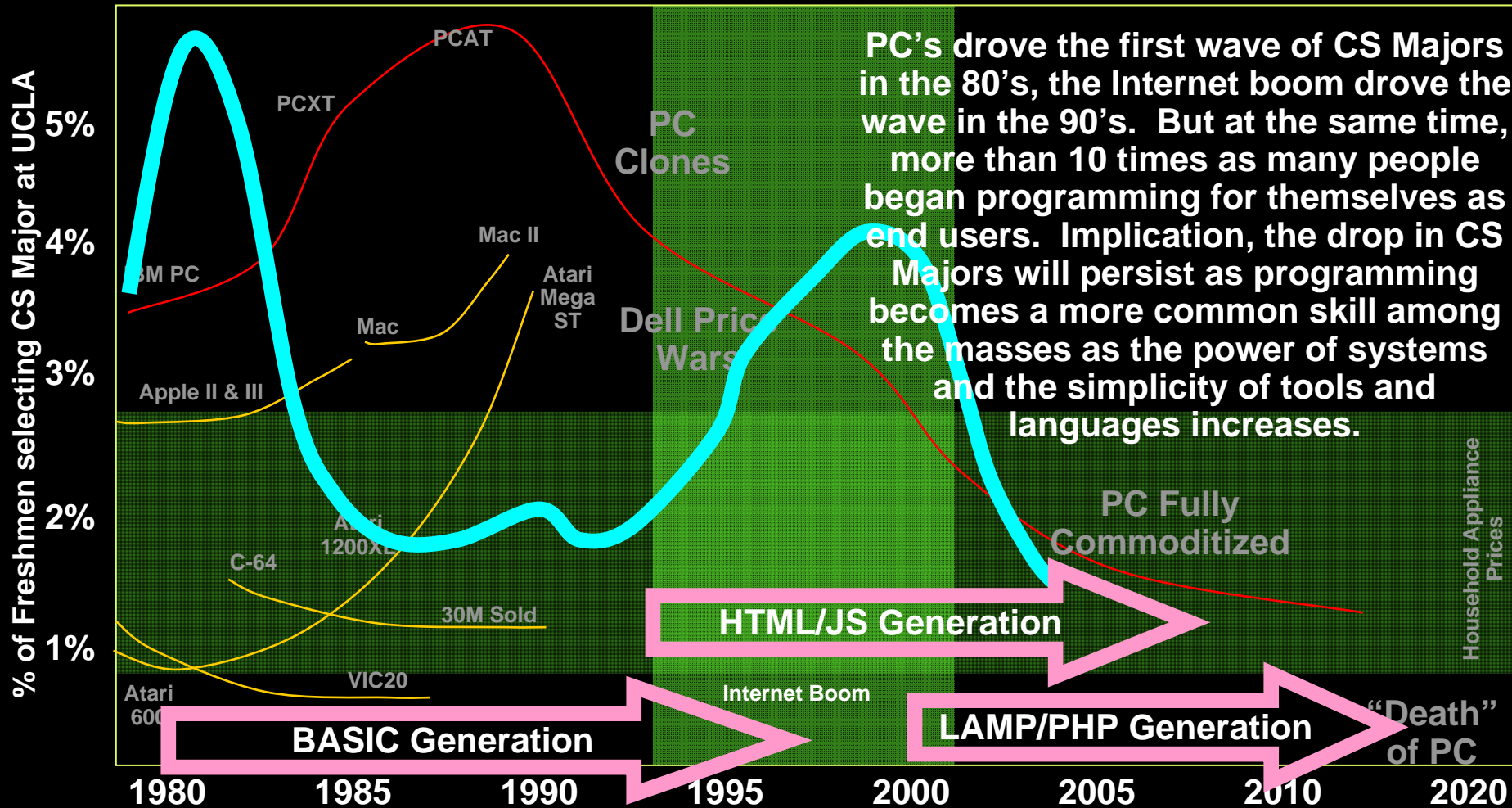


Sources: [www.old-computers.com](http://www.old-computers.com)



# Evolution of Programmers –The BASIC Generation

“My first class in computer programming was an elective course in BASIC back in sixth grade.”

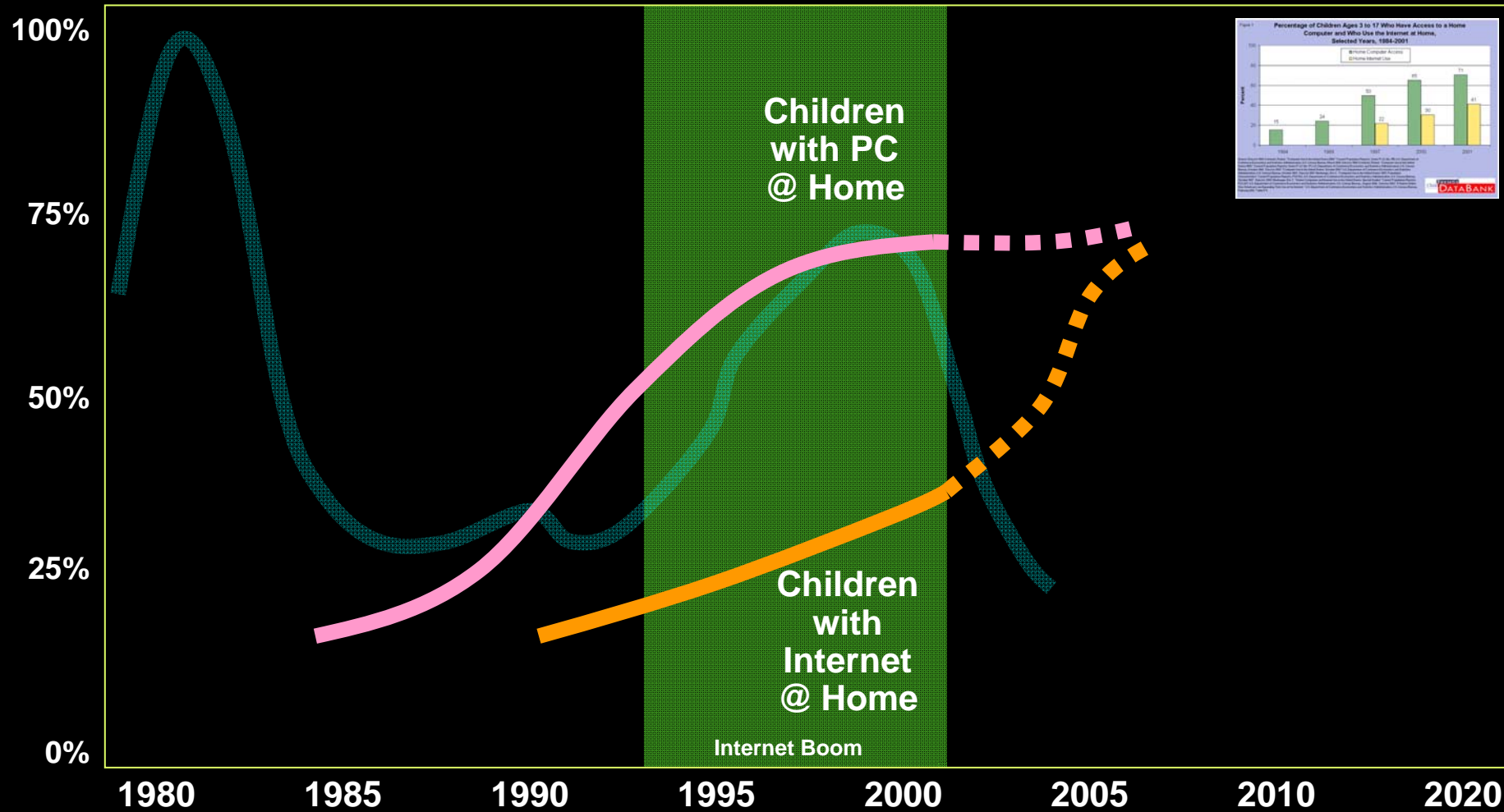


PC's drove the first wave of CS Majors in the 80's, the Internet boom drove the wave in the 90's. But at the same time, more than 10 times as many people began programming for themselves as end users. Implication, the drop in CS Majors will persist as programming becomes a more common skill among the masses as the power of systems and the simplicity of tools and languages increases.

Sources: www.old-computers.com, May 2005 Computing Research News



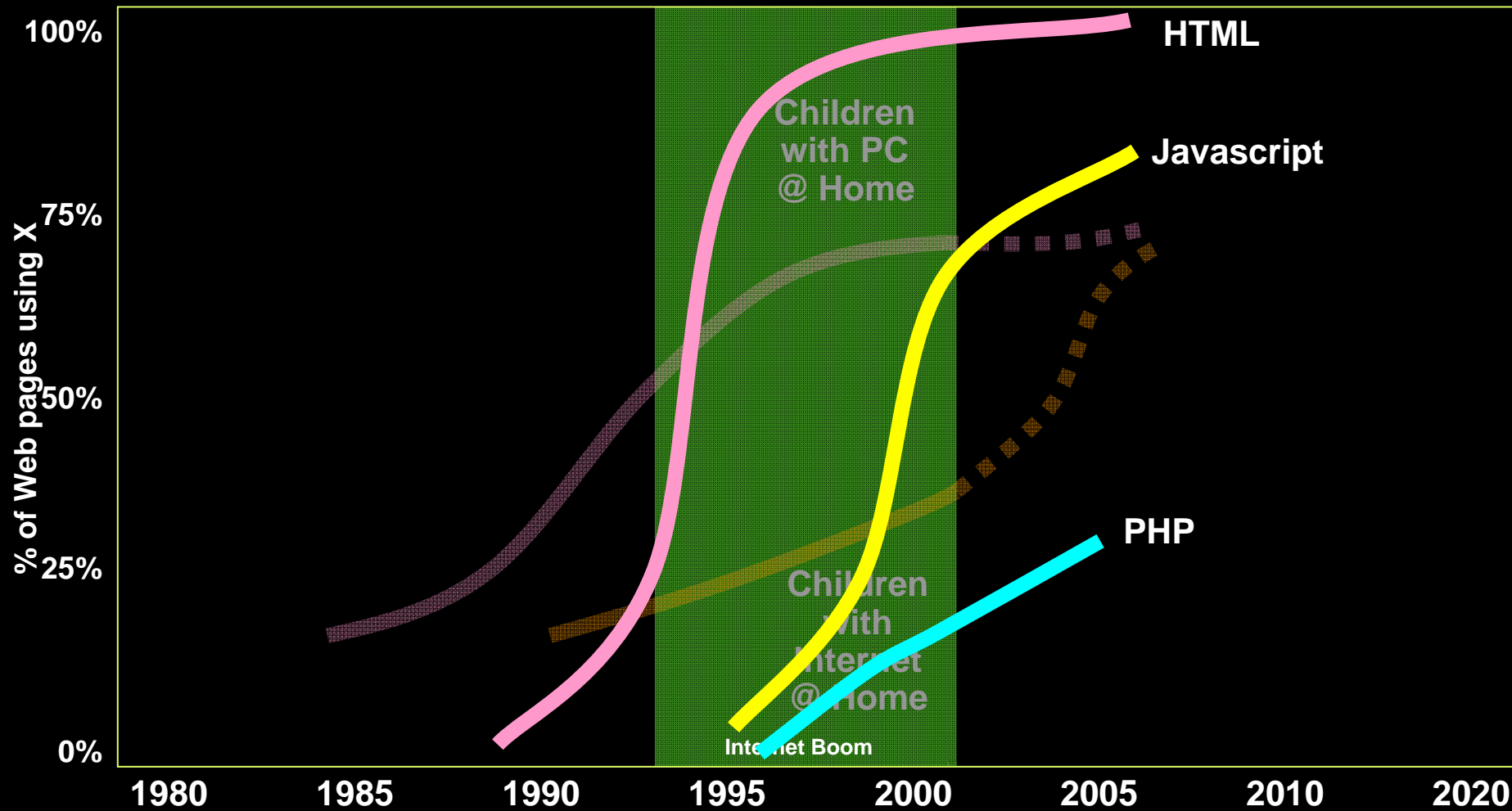
# Evolution of Programmers – Children PC/Internet Trend (US)



Sources: [www.old-computers.com](http://www.old-computers.com), May 2005 Computing Research News, [www.childtrendsdatbank.org](http://www.childtrendsdatbank.org)



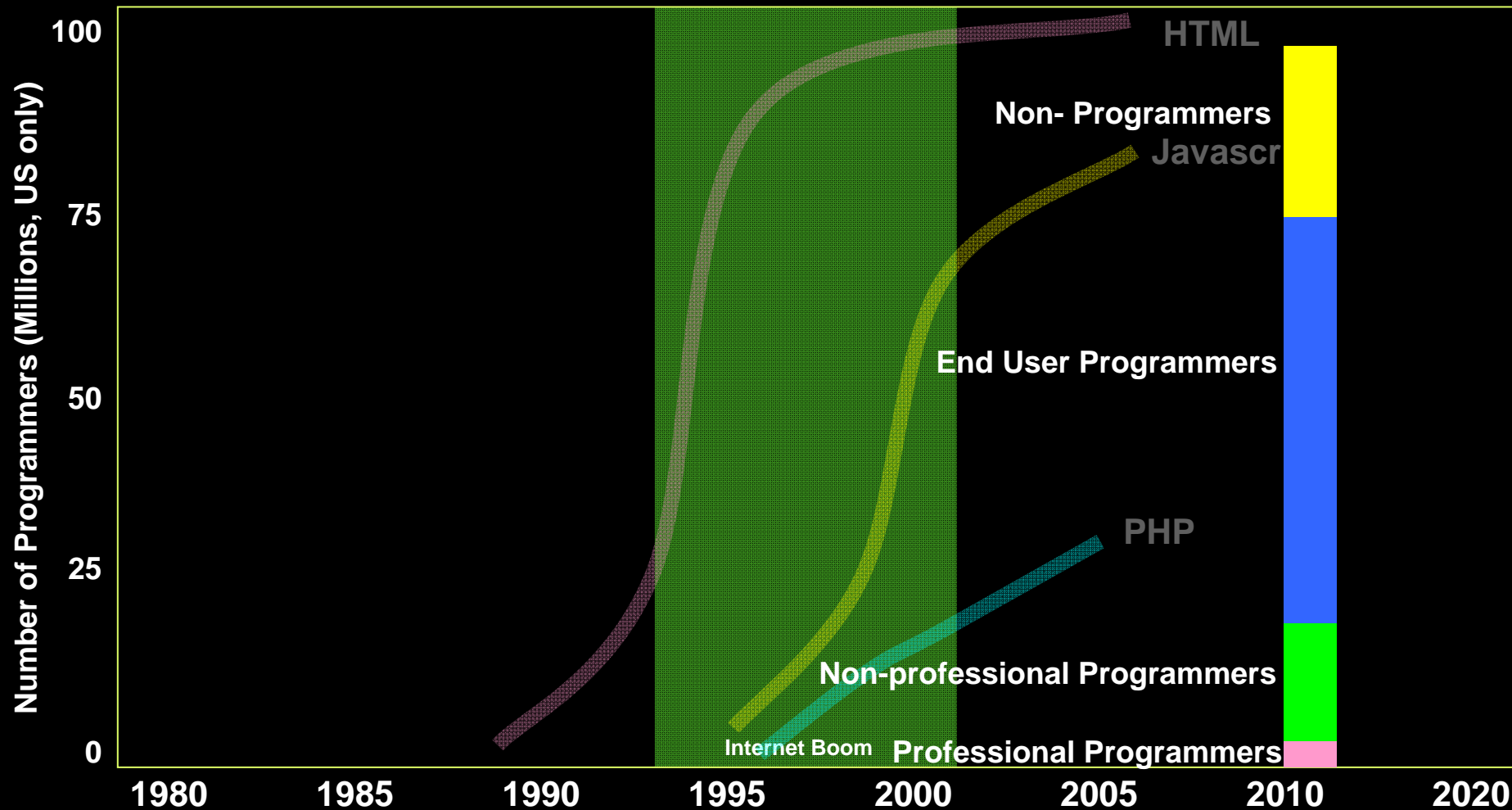
# Evolution of Programmers – Web Scripting Trend



Sources: [www.old-computers.com](http://www.old-computers.com), May 2005 Computing Research News, [www.childtrendsdatbank.org](http://www.childtrendsdatbank.org)



# Evolution of Programmers – Non-Professional Trend



Sources: Estimating the Numbers of End Users and End User Programmers, Scaffidi, Shaw, and Myers, VL/HCC, 2005



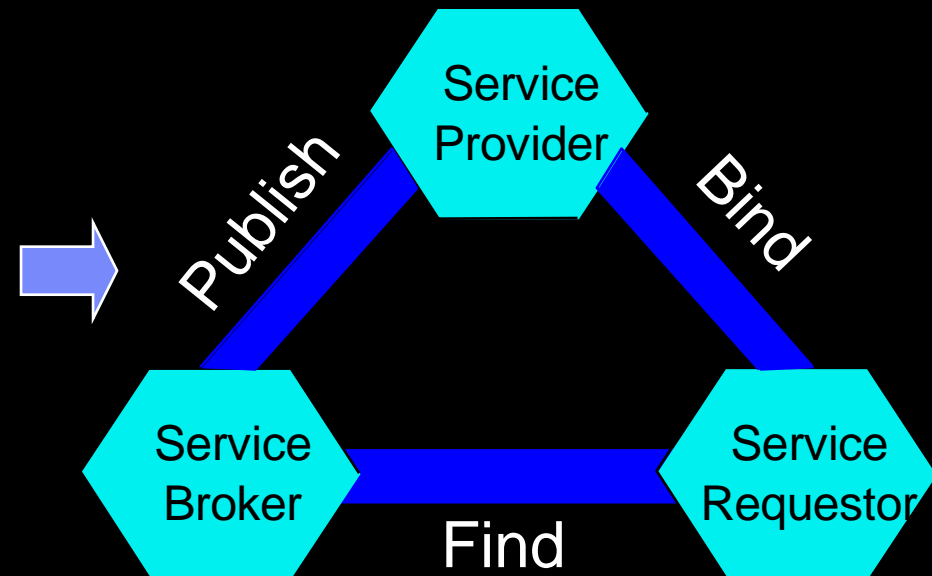
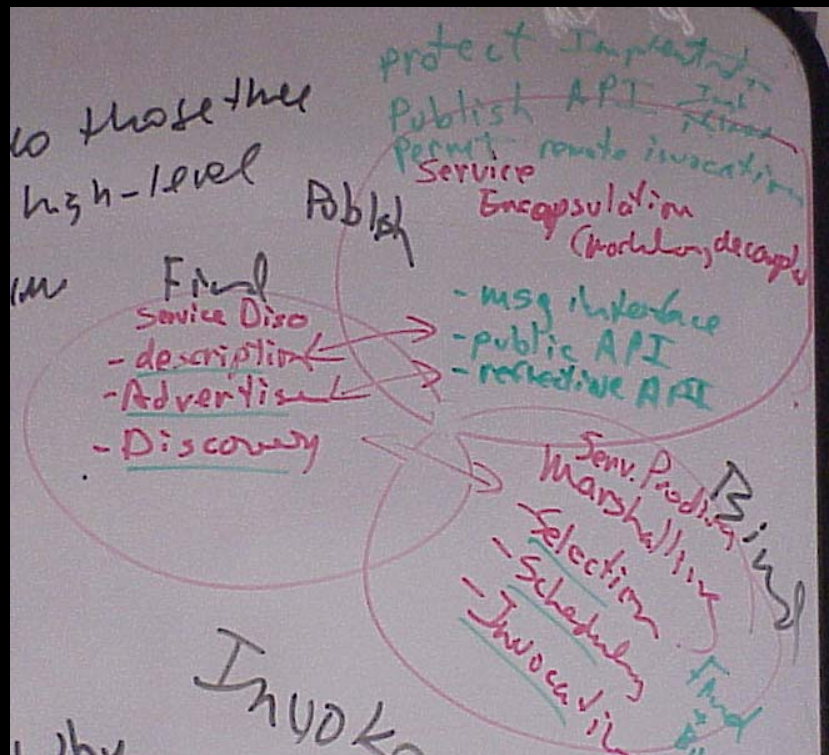
# Megatrends

- ❖ **Generational change and expectations**
- ❖ **The evolution of abstraction and infrastructure**
- ❖ **Web 2.0 and the real Open Source Movement**
- ❖ **The Really, Really Long Tail of Software**
- ❖ **Technology evolution and End User Programming waves**
- ❖ **The coming massively multicore software (r)evolution**



# A little perspective: In the Beginning...

Adams/Graham, November 1998



Steve: So does that mean we have, "service oriented programming"?

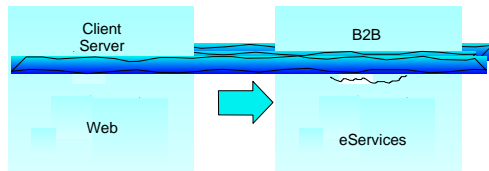
Sam: Nope, That's an architecture, a "service oriented architecture"



# The origin of SOA, circa late 1998

## SOA Paradigm Shift

- The next generation of e-business will be "service oriented"
- This will cause a big change in IBM's world



## Ubiquitous Protocols Drive Change

- Every new widely adopted protocol layer ushers in a paradigm shift in distributed computing
  - ▶ TCP/IP adoption drove Client/Server
  - ▶ HTTP adoption drove Browser/WAS
  - ▶ "eSP" adoption will drive a similar large scale change
    - eSP: eServices Protocol: XML-based protocol for publishing, discovering and invoking services

Protocol	Architecture	Dist. Comp. Model
TCP/IP	Client/Server	CORBA/COM/DCOM
HTTP	Browser/WAS	servlets/beans/JSP/ASP/EJB
eSP	SOA ePlatform	"services"

## Shift Architecture - Accelerate e-business Growth

- Client-server
    - ▶ Innovation generated by building and distributing monolithic applications and packaged software
  - Browser/WAS
    - ▶ Innovation generated by integration of back end systems and web-facing front ends
  - SOA
    - ▶ Integration of existing services generates new innovation
- IMPORTANT: Each successful architecture was constructed using the previous architecture, SOA is a natural extension to our existing WAS platform

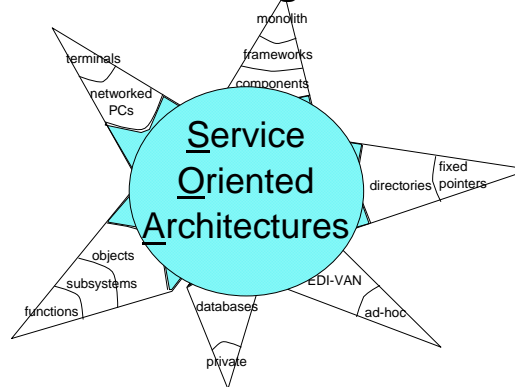
Architecture	Growth Model	Time Scale
Client/Server	innovate -> populate	months/years
Browser/WAS	innovate -> integrate	weeks/months
SOA	integrate -> innovate	hours/days

## Decoupling to Maximize Reach

- e-business value proposition
  - ▶ **Maximize** reach to customers, choice of suppliers/partners, flexibility of business processes, speed of execution
  - ▶ **Minimize** time to market, cost
- Single technical strategy: Decouple, encapsulate, then reintegrate using higher level messages

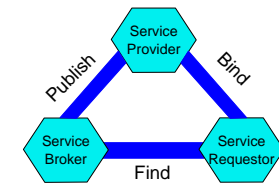
	Coupling		
	Tight		Loose
Messaging	ad-hoc	traditional EDI	XML B2B
Componentry	Functions	Subsystems Objects	Services
Data	Private format	SQL	XML
UI	Inline	MVC	XML Xcoding
Package/Deploy	Monolith	Frameworks Components	Services
Directory	Pointers (none)	LDAP	Dyn. Discov.

## Where is this decoupling trend leading?



## SOA Defined

- 3 Roles
  - ▶ Service Provider, Service Requestor, Service Broker
- 3 Operations
  - ▶ Publish, Find, Bind
- Operations constrained by pluggable Intermediaries



- Requires widely adopted Well Defined Services & eServices Protocol



# Decoupling to Maximize Reach

- **e-business value proposition**
  - ▶ **Maximize** reach to customers, choice of suppliers/partners, flexibility of business processes, speed of execution
  - ▶ **Minimize** time to market, cost
- Single technical strategy: Decouple, encapsulate, then reintegrate using higher level messages

	Coupling			
	Tight			Loose
Messaging	ad-hoc	traditional EDI		XML B2B
Componentry	Functions	Subsystems	Objects	Services
Data	Private format		SQL	XML
UI	Inline	MVC		XML Xcoding
Package/Deploy	Monolith	Frameworks	Components	Services
Directory	Pointers (none)		LDAP	Dyn. Discov.



# Megatrends

- ❖ **Generational change and expectations**
- ❖ **The evolution of abstraction and infrastructure**
- ❖ **Web 2.0 and the real Open Source Movement**
- ❖ **The Really, Really Long Tail of Software**
- ❖ **Technology evolution and End User Programming waves**
- ❖ **The coming massively multicore software (r)evolution**



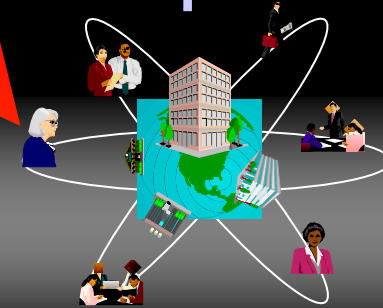
## Web 2.0

- **Is less about systems and more about people**
- **Less about pages/documents and more about fragments**
- **Less about apps and more about services**
- **Less about portals and more about feeds**
- **Less about windows and more about widgets**
  
- **And is fundamentally about end user creation of content and function**

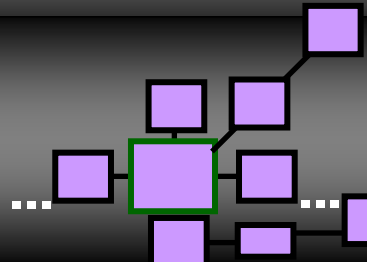


# Social and Technical Trends are Accelerating the Enterprise into the Future

Enterprise 2.0 - realized



SOA - realized



- REST-based interfaces simplifying integration
- **User-made Mashups/Situational Apps abound**
- Personal computing migrating to multi-device
- Personal Memory Devices enabling digital persona
- Cheap hi-res screens enabling "ambient" displays
- Wireless broadband bringing everything "online"
- Widgets replacing Windows as the GUI metaphor
- Real-time translation breaks the language barrier

- Social computing /networking
- Tag-based search
- **From participatory media (wikis/blogs) to participatory development (widgets/mashups)**
- Younger generations driving adoption

**Normal Trajectory**

**Technology Trends**

**Social Trends**



# “World Wide Widgets”

- ❖ The emerging post-windows, post-browser interface metaphor
- ❖ Limited widget size and few function points support end user assembly
- ❖ Requires “micro” mindset (microcontexts, microtemplates, microformats)



Early applications assumed they owned the entire display

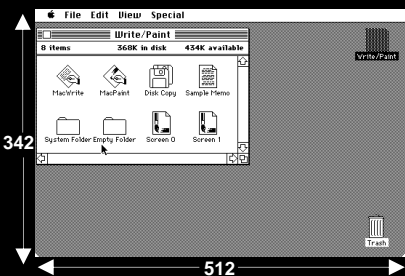
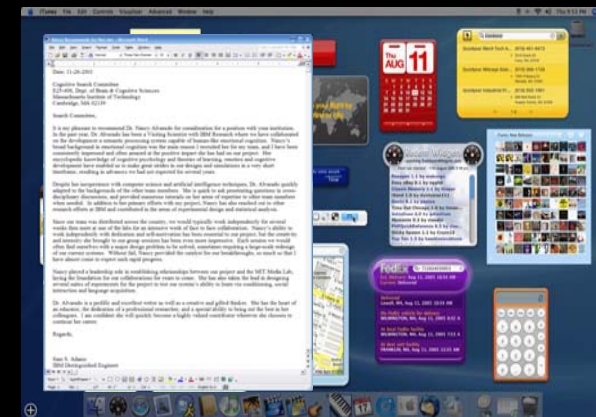


Until recently, users routinely maximized their current application to use the full screen



With the advent of 2 megapixel displays, users no longer routinely maximize apps

Extra screen area is now used for persistent mini-apps, a.k.a., desktop widgets



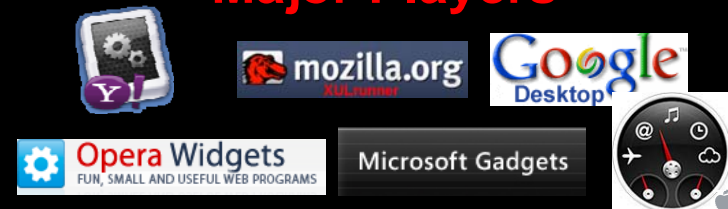
Icon/Folder metaphor worked well for early low-rez displays



Icons become useless unreadable clutter on modern hi-rez displays



## Major Players





# “World Wide Widgets”

- ❖ A truly cross device graphical interface metaphor
- ❖ A typical widget is just the right size for a cell phone display and simple for a kiosk or touchscreen





# SW Eng issues for the widget world

## ■ Issues

- ❖ How can users develop their own?
- ❖ How to manage groups of widgets?
- ❖ How to share them collaboratively?
- ❖ How to mashup desktop widgets?
- ❖ How to update them? Feeds?
- ❖ What about standards? Interoperability?
- ❖ Single sign-on?
- ❖ Security?

**“Portal” infrastructure requirements don’t go away**



# The real Open Source Movement

- **The real Open Source Movement did not begin with Stallman and Raymond**
- **Its started in the large with End Users sharing BASIC and Spreadsheets**
- **What is the most powerful invention for Open Source (and End User Programming) in the past decade?**

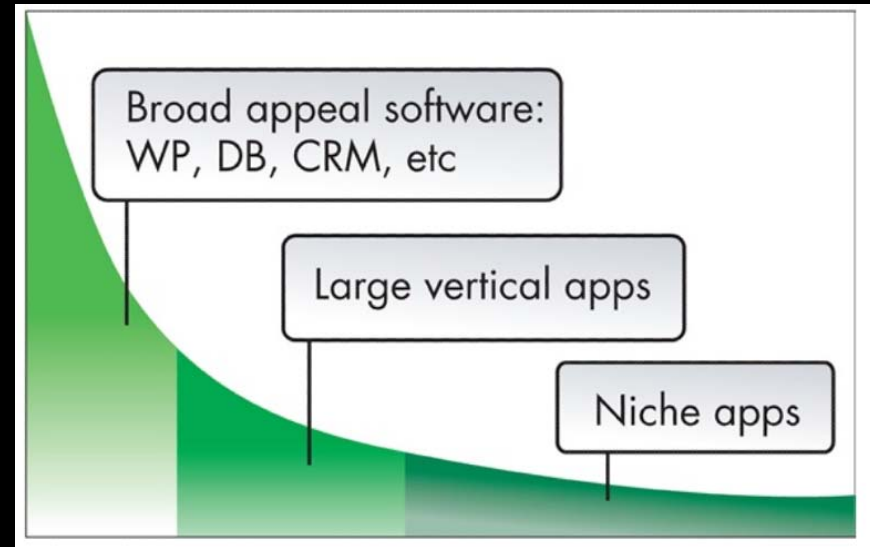
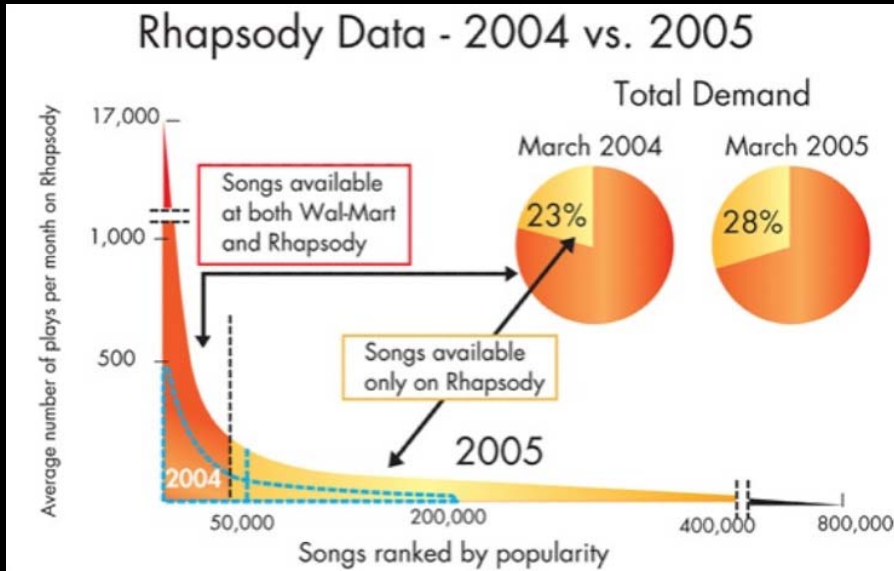


# Megatrends

- ❖ **Generational change and expectations**
- ❖ **The evolution of abstraction and infrastructure**
- ❖ **Web 2.0 and the real Open Source Movement**
- ❖ **The Really, Really Long Tail of Software**
- ❖ **Technology evolution and End User Programming waves**
- ❖ **The coming massively multicore software (r)evolution**



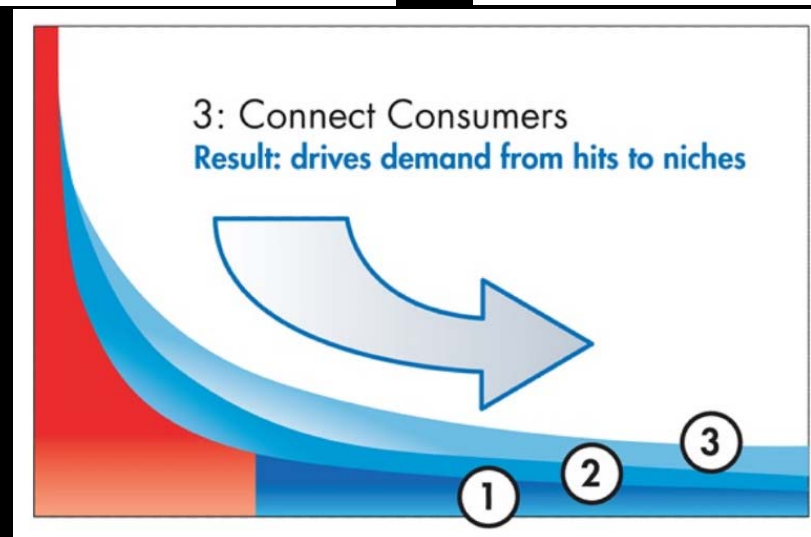
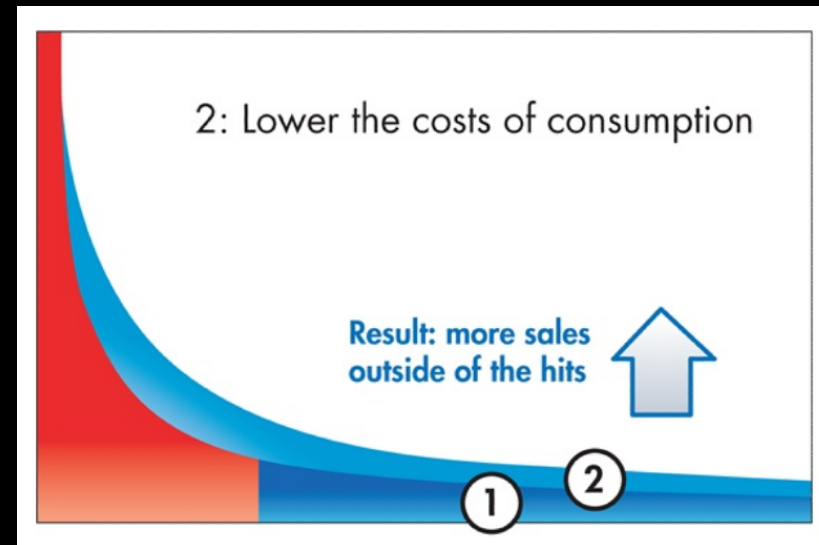
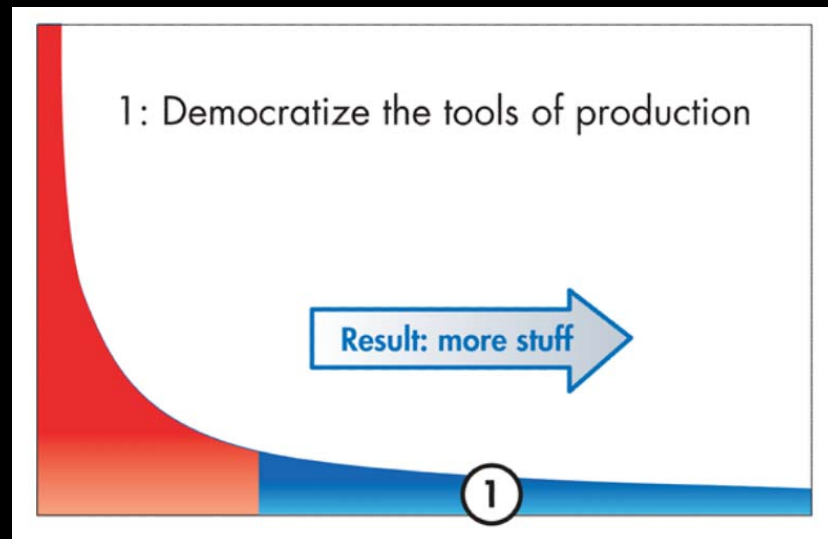
# Understanding the Long Tail of Software Development



Source: <http://salesforce.breezecentral.com/longtail/>



# The Three Forces of the Long Tail



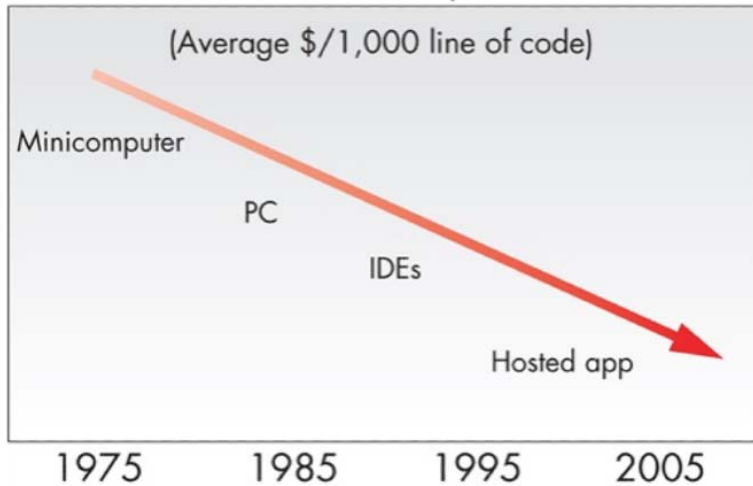
Source: <http://salesforce.breezecentral.com/longtail/>



# Applying these forces to software development

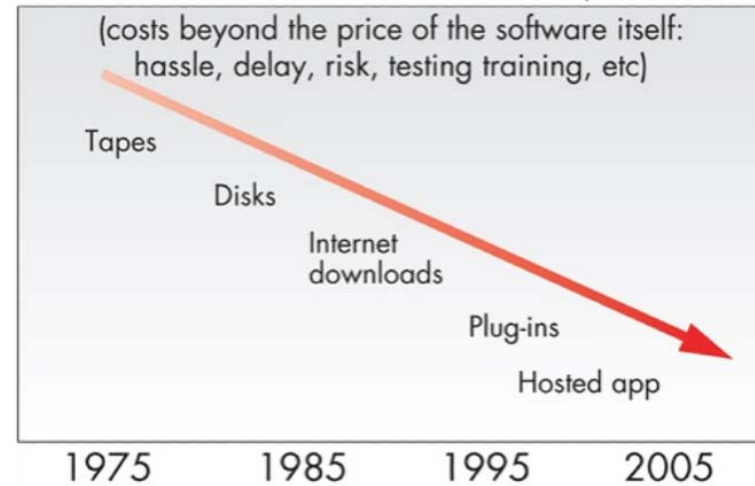
### Force 1: Cost of production

(Average \$/1,000 line of code)



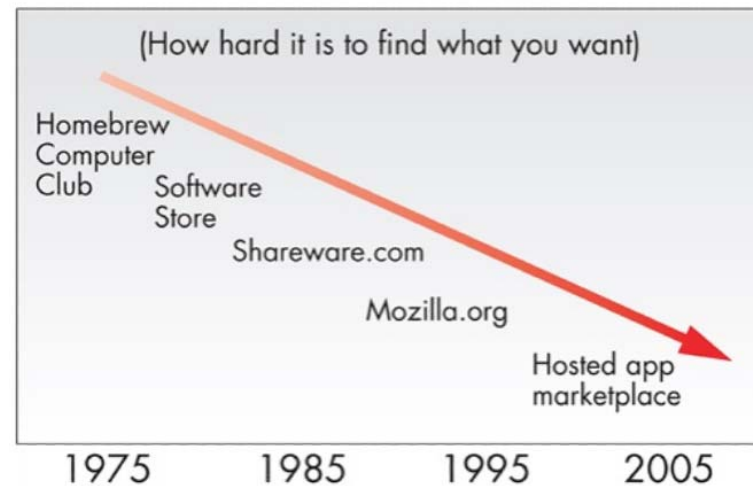
### Force 2: Cost of consumption

(costs beyond the price of the software itself: hassle, delay, risk, testing training, etc)



### Force 3: Search costs

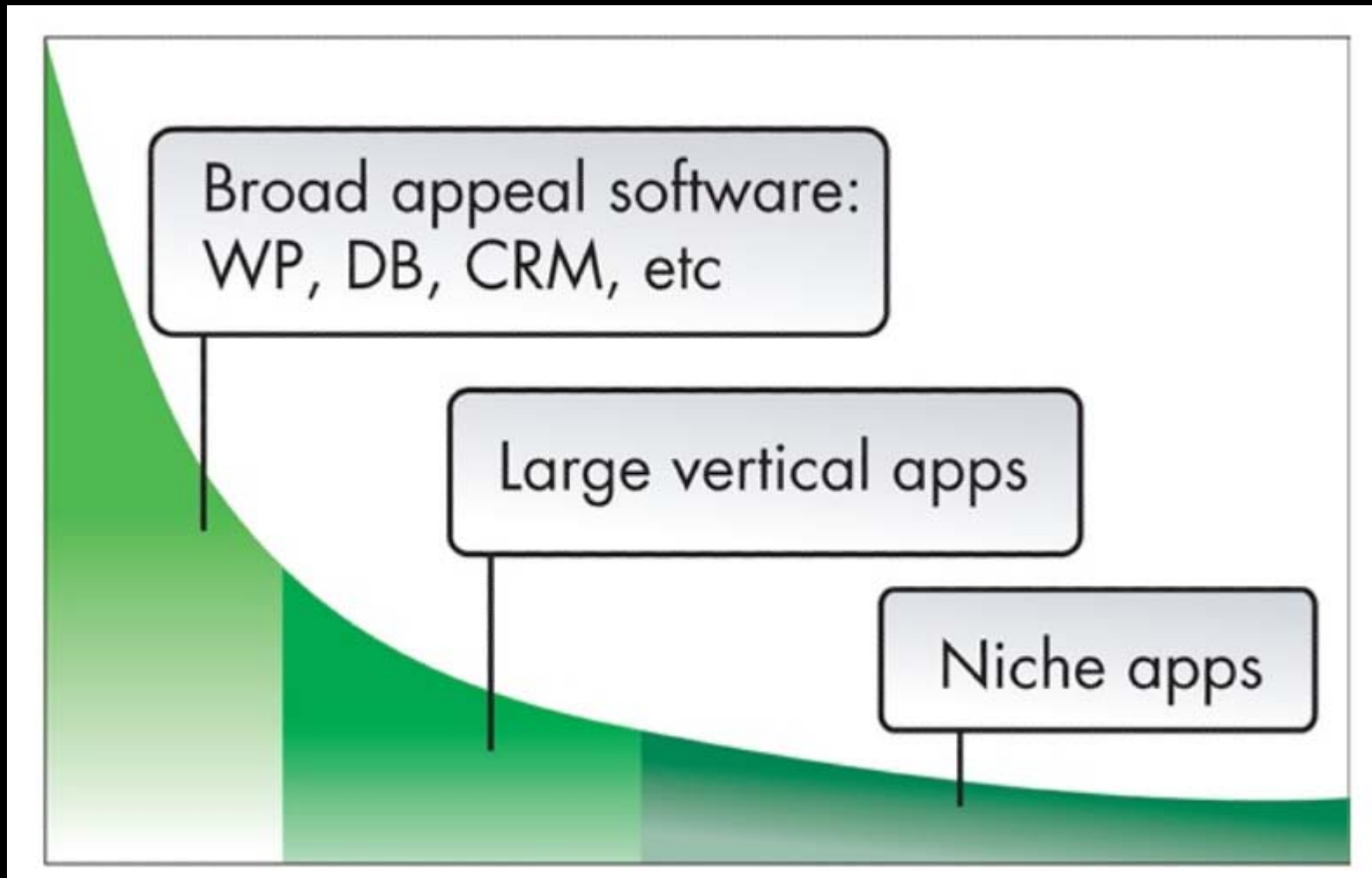
(How hard it is to find what you want)



Source: <http://salesforce.breezecentral.com/longtail/>



## And you get the Long Tail of Software



**But, this model still assumes some level of professional dev skill**

Source: <http://salesforce.breezecentral.com/longtail/>



# Inverting the marketplace for software

## ■ Past

- ❖ Dozens of markets (software product categories) of millions (users)
- ❖ Mass market offerings, boxed \$oftware, bricks-n-mortar stores, PC preloads
- ❖ 10s to 1000s of professional developers per product

## ■ Present

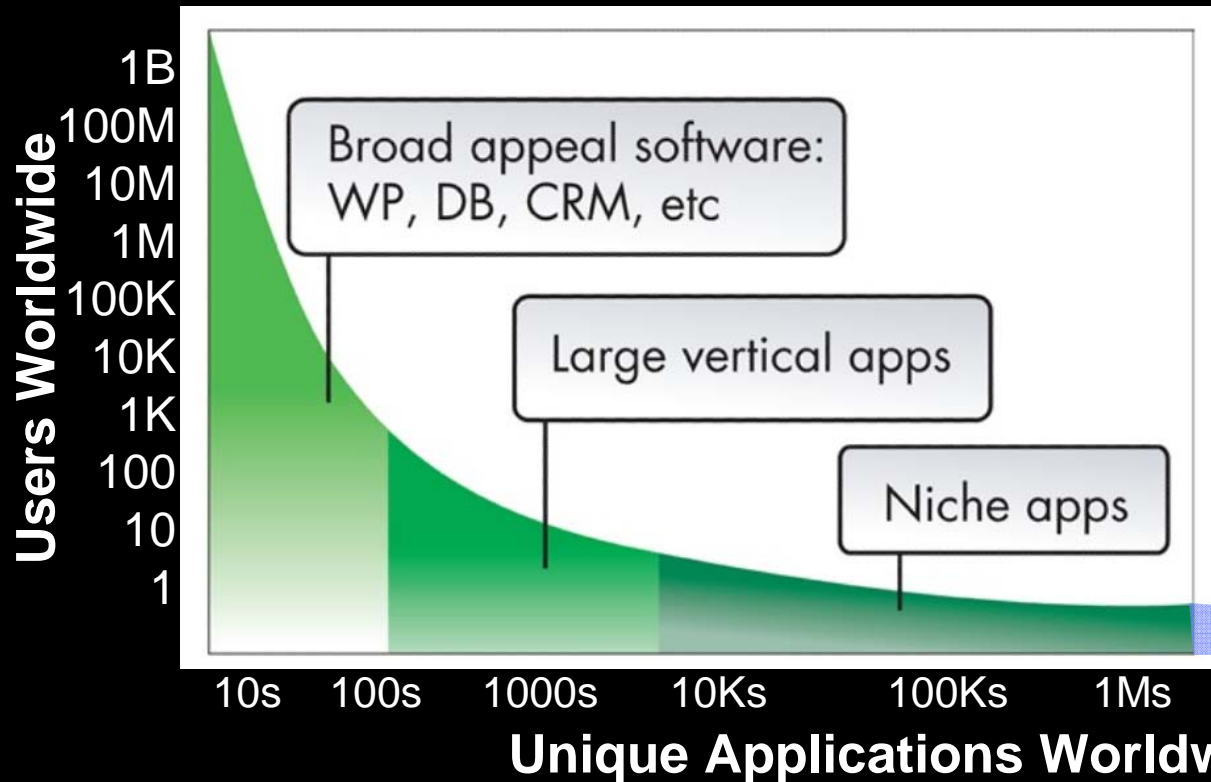
- ❖ “Millions of markets of dozens” – Joe Kraus, CEO JotSpot, now Google
- ❖ Current Web model, niche offerings, downloaded, \$oftware as \$ervice
- ❖ 1s to 100s of professional developers, some end user customization

## ■ Future

- ❖ Billions of markets of 1
- ❖ Large applications/suites dissolve into dashboards of widgets, extreme customization and personalization
- ❖ No professional developers required, assembled from widgets by users for personal use



# Where is the rest of the Long Tail of Software?



**End User Programming  
of the user  
by the user  
for the user**



# Megatrends

- ❖ **Generational change and expectations**
- ❖ **The evolution of abstraction and infrastructure**
- ❖ **Web 2.0 and the real Open Source Movement**
- ❖ **The Really, Really Long Tail of Software**
- ❖ **Technology evolution and End User Programming waves**
- ❖ **The coming massively multicore software (r)evolution**



# Megatrends

- ❖ **Generational change and expectations**
- ❖ **The evolution of abstraction and infrastructure**
- ❖ **Web 2.0 and the real Open Source Movement**
- ❖ **The Really, Really Long Tail of Software**
- ❖ **Technology evolution and End User Programming waves**
- ❖ **The coming massively multicore software (r)evolution**



# DO

- **What can the Software Engineering community do to help?**



# Implicit Software Engineering for EUP

- **Unburden the End User Programmer**
- **In fact, PLEASE unburden the Professional Programmer!**
- **We don't all have to think like CPUs**
  
- **At ICSE 2028, do we really still want to be struggling with loop index and other “of by one” errors?**
- **A huge proportion of professional programming errors are still boundary issues partially arising from primitive numeric types**
- **Can we get integers (ala Pythagoras) right this time?**



## Explicit SW Eng for End User Programmers

- **Help create more concrete, domain-specific languages instead of YAGPPL**
- **Help EUPs deal with the rest of the lifecycle**
- **Debugging, testing, version/release management**
- **Security, provenance, governance**
- **EUSE wants you!**



# Include

- **<women.h>**
  - ❖ **More programmers means more women programmers**
  - ❖ **Fascinating work on gender differences in programming**
  
- **<non-english-speakers.h>**
  
- **<non-rich-1st-worlders.h>**
  - ❖ **OLPC is a good step**
  - ❖ **How about empowering them all to be end user programmers?**



# HERESY!

- **WASTE something to benefit End User Programmers**
- **Alan Kay's vision of Personal Computing assumed**
  - ❖ 1M cycles per second
  - ❖ 1M bytes of RAM
  - ❖ 1M pixels
- **Remember WIMP attitude?**
- **What happens in a single processor is beginning to matter about as much as what happens in a single biological cell.**
  - ❖ Multicellular Computing, [evolutionofcomputing.org](http://evolutionofcomputing.org)
- **What will you “waste” to empower future end users?**
  - ❖ A terraflop?
  - ❖ 1M cores?



**The future of SW Engineering is in your hands**

**IF ( happens(nothing) )**

**{ happens(nothing) }**

**ELSE**

**{<< your future here >>}**

**ENDIF**