IBM®

RU READY TO SAVE THE DAY

IBM Rational Software Development Conference 2008

WHERE TEAMS ARE R-HEROES

IBM Rational Software Development Conference 2008

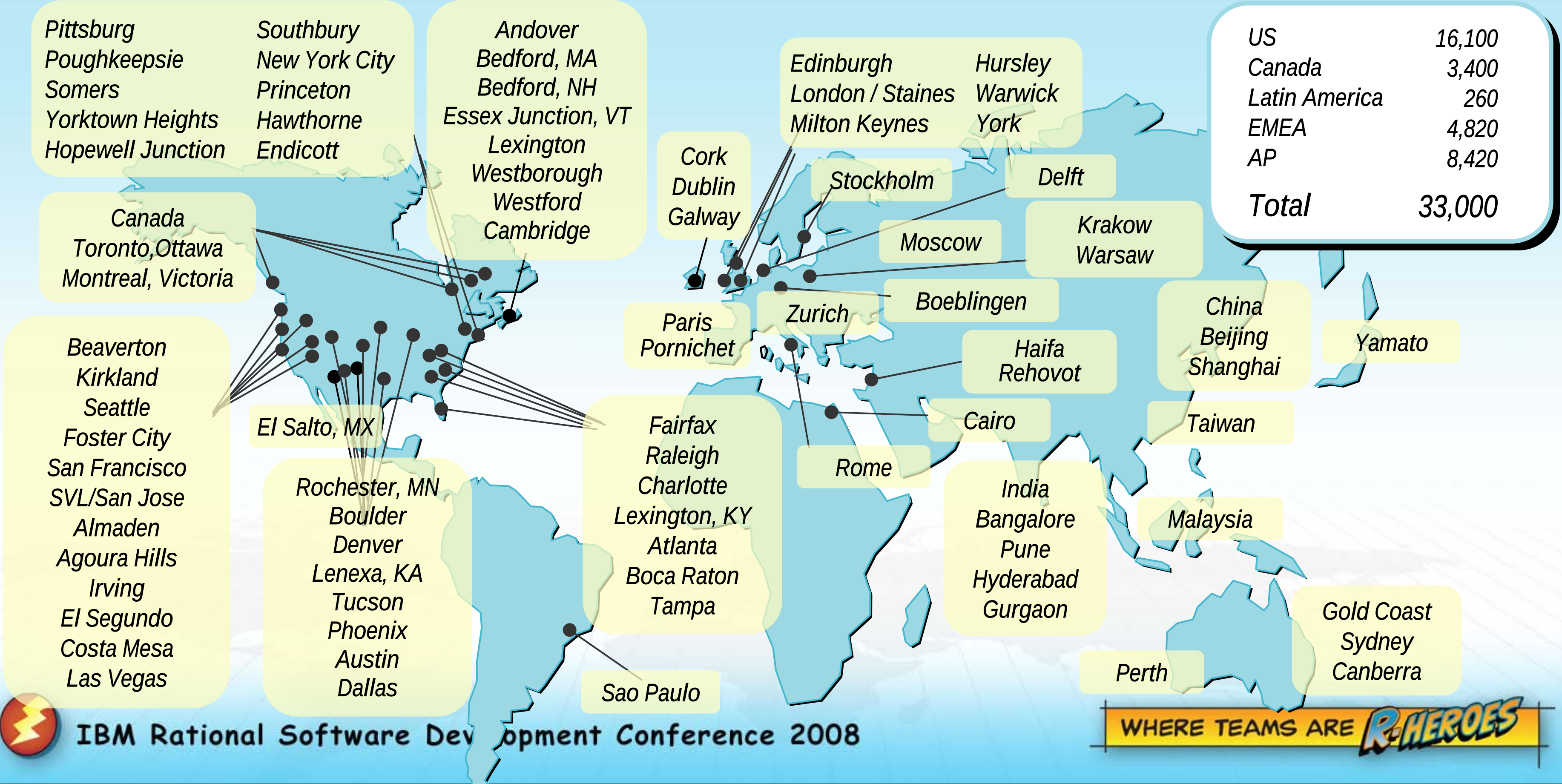
WHERE TEAMS ARE R-HEROES

SCOTT HEBNER

VICE PRESIDENT
MARKETING & STRATEGY
IBM RATIONAL SOFTWARE

Software Development in IBM

A Global Team of IBM Software Developers



Pittsburg
Poughkeepsie
Somers
Yorktown Heights
Hopewell Junction

Southbury
New York City
Princeton
Hawthorne
Endicott

Andover
Bedford, MA
Bedford, NH
Essex Junction, VT
Lexington
Westborough
Westford
Cambridge

Canada
Toronto,Ottawa
Montreal, Victoria

Beaverton
Kirkland
Seattle
Foster City
San Francisco
SVL/San Jose
Almaden
Agoura Hills
Irving
El Segundo
Costa Mesa
Las Vegas

El Salto, MX

Rochester, MN
Boulder
Denver
Lenexa, KA
Tucson
Phoenix
Austin
Dallas

Fairfax
Raleigh
Charlotte
Lexington, KY
Atlanta
Boca Raton
Tampa

Sao Paulo

Cork
Dublin
Galway

Edinburgh
London / Staines
Milton Keynes

Hursley
Warwick
York

Stockholm

Delft

Moscow

Krakow
Warsaw

Paris
Pornichet

Zurich

Boeblingen

Haifa
Rehovot

Cairo

Rome

India
Bangalore
Pune
Hyderabad
Gurgaon

China
Beijing
Shanghai

Yamato

Taiwan

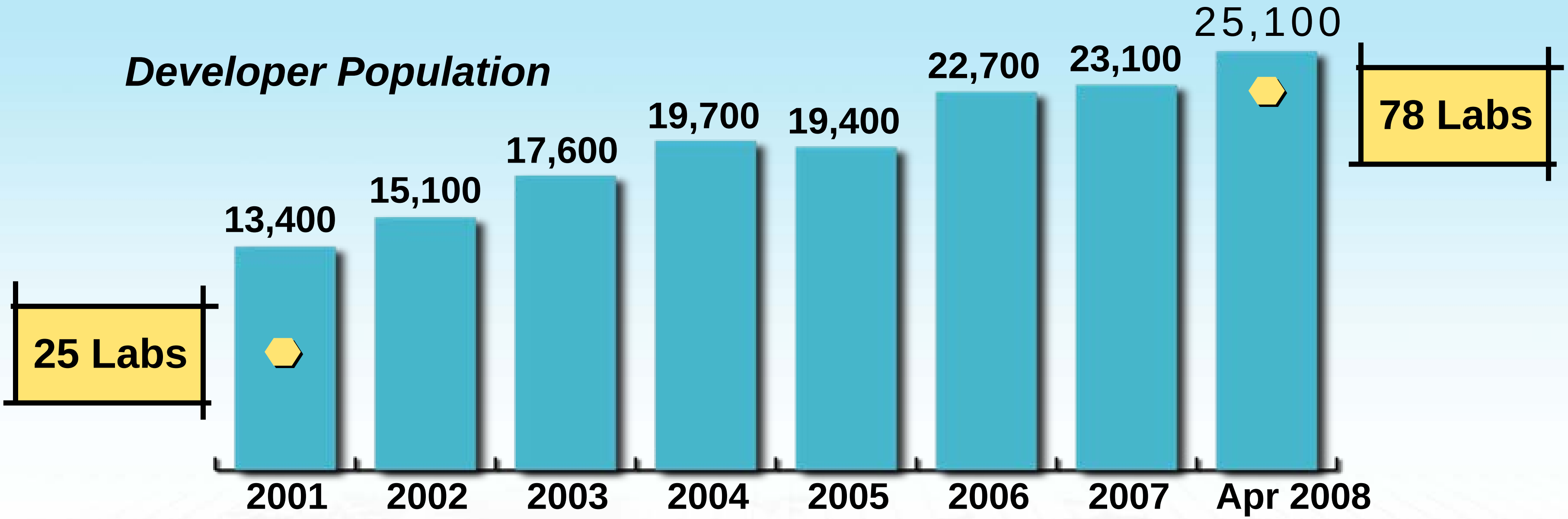
Malaysia

Perth

Gold Coast
Sydney
Canberra

US	16,100
Canada	3,400
Latin America	260
EMEA	4,820
AP	8,420
Total	**33,000**

IBM Software Group Developer Community Growth

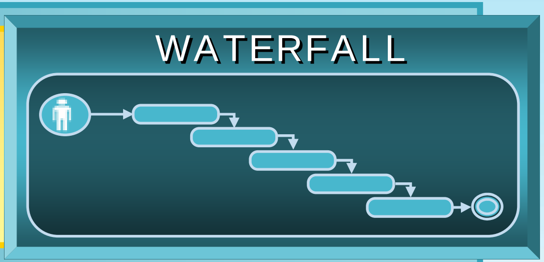


- 7,527 developers from 54 acquisitions since 2001
- 4,200 developers through organic growth
- Acquired and retained 40 Lab locations world-wide since 2001

IBM Software Development Transformation

1980's

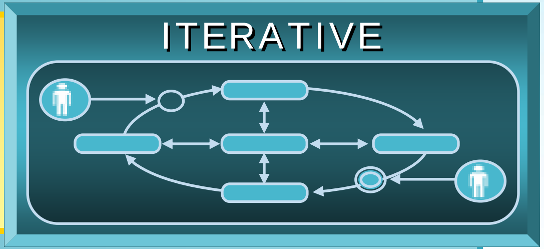
- Waterfall development
  - Rigid, late feedback, slow reaction to market changes


WATERFALL

Rigid

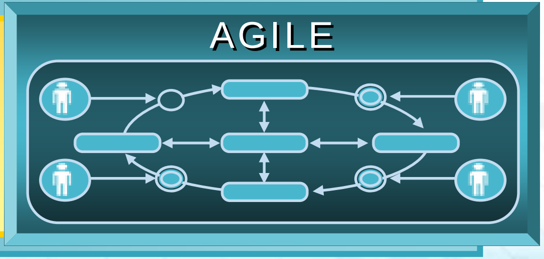
1990's

- Iterative development
  - Customized RUP, community source and component reuse, emphasis on consumability


ITERATIVE

Present

- Agile / Lean development
  - Global reach, SOA, agile practices, outside-in development, tools and not rules


AGILE

Continuous Learning and Adaptive Planning

Development Governance Principles

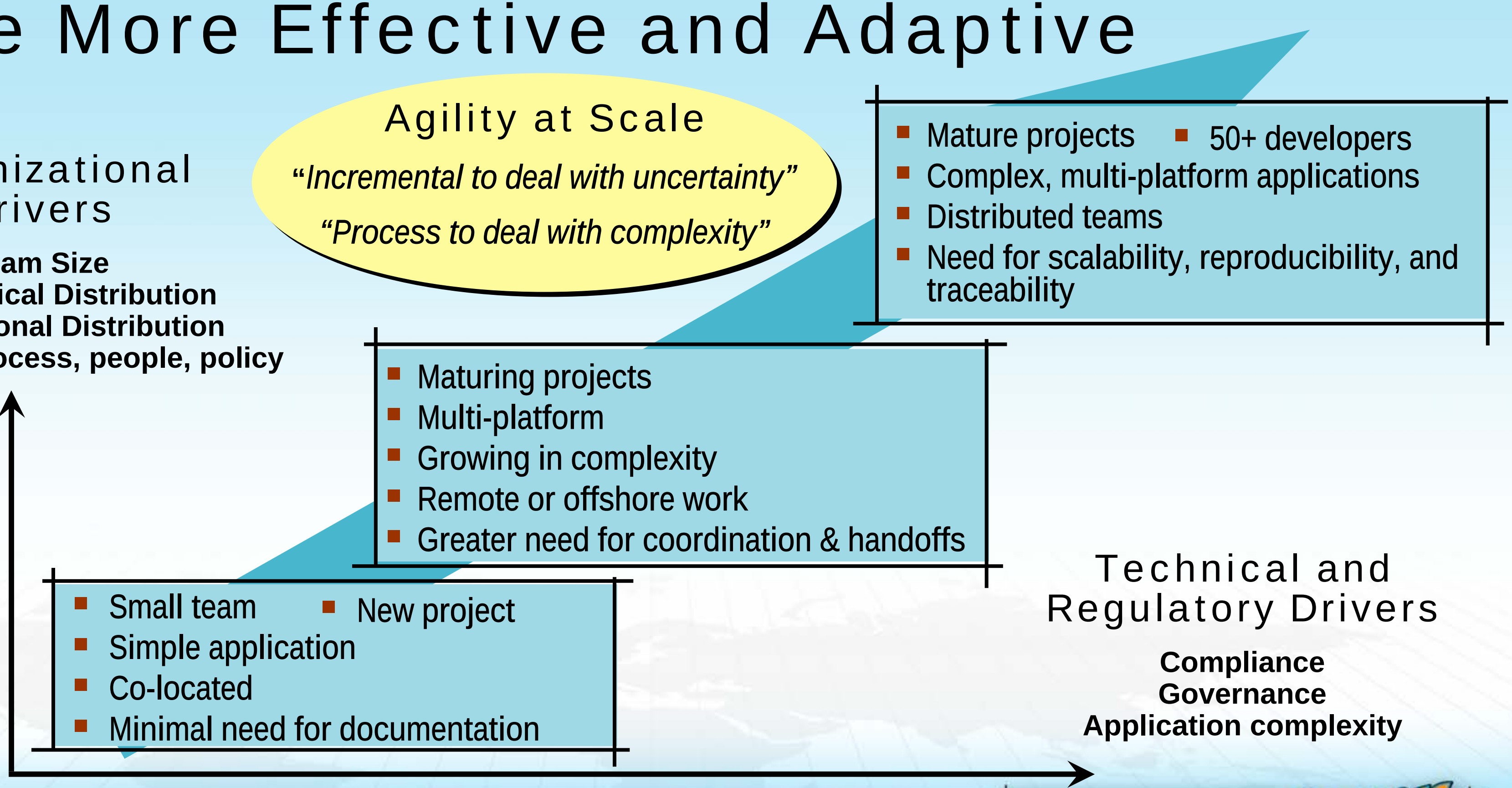
Lightweight central mechanisms

Tools, *not* *Rules*

Support for Communities

Centralized development services

Diversity and Complexity Requires Teams to be More Effective and Adaptive



Agility at Scale

"Incremental to deal with uncertainty"

"Process to deal with complexity"

Organizational Drivers

Team Size
Geographical Distribution
Organizational Distribution
Entrenched process, people, policy

- Small team
- New project
- Simple application
- Co-located
- Minimal need for documentation

- Maturing projects
- Multi-platform
- Growing in complexity
- Remote or offshore work
- Greater need for coordination & handoffs

- Mature projects
- 50+ developers
- Complex, multi-platform applications
- Distributed teams
- Need for scalability, reproducibility, and traceability

Technical and Regulatory Drivers

Compliance
Governance
Application complexity

Development Transformation – Driving Change

Outside-in Design

Agile Development

Componentization and Reuse

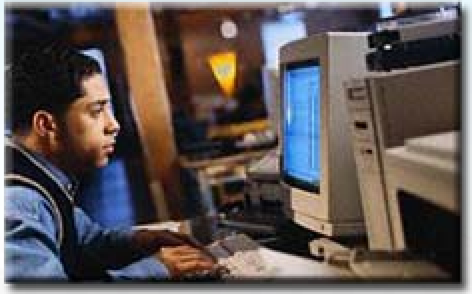
Communities and Community Source

Outside-In Development

Outside-in development is about focusing on the business stakeholders who are affected by your software, and about applying that focus to the entire software cycle

The Four "Must Do's"

1. Focus on the stakeholders
2. Develop business scenarios
3. Use iterations
4. Remember that both iterations and final product need to be consumable by the target stakeholders and users



Agile Software Engineering

- Iterative, typically time-boxed as short iterations
- About frequent, even constant, validation with stakeholders
- Highly focused on mitigating risks
- Adaptive; comfortable with change & reprioritization
- Communication intensive (e.g., daily Scrums)
- Aimed at making incremental progress; working software is the measure
- Disciplined, scaleable, and workable across sites

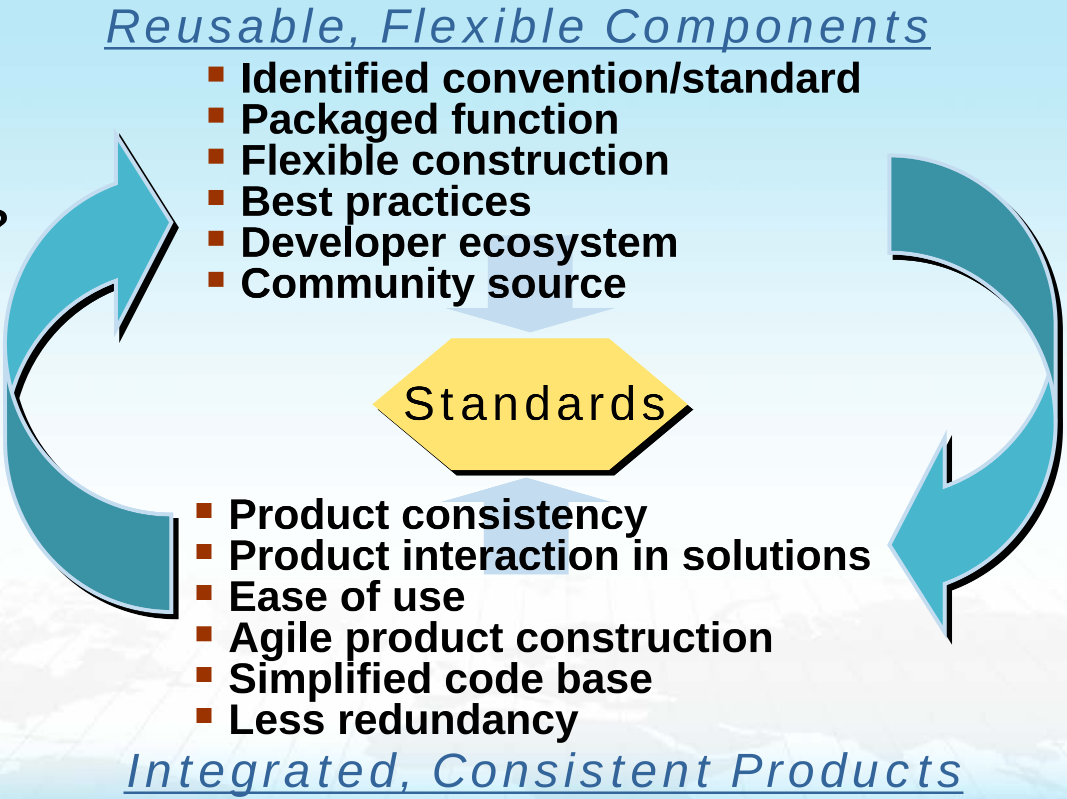
A good agile project will build something that meets customers needs but may be different from original plans

Components for Product Integration and Simplification

What integration problem are we trying to solve?

What components do we need?

What conventions exist or do we need to develop?



Reusable, Flexible Components

- Identified convention/standard
- Packaged function
- Flexible construction
- Best practices
- Developer ecosystem
- Community source

Standards

- Product consistency
- Product interaction in solutions
- Ease of use
- Agile product construction
- Simplified code base
- Less redundancy

Integrated, Consistent Products

What conventions are supported or implied?

When/where do we drive these Into products?

Do we have components that fit or are we creating some? (through re-factoring)

Component Reuse in IBM

WebSphere Application Server **technology has been reused and/or bundled with** *126* **other product offerings**

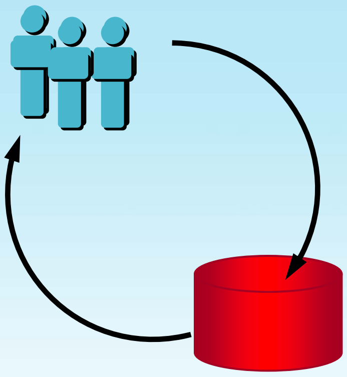
DB2 **technology has been reused and/or bundled with** *175* **other product offerings**

WebSphere Portal **technology has been reused and/or bundled with** *15* **other product offerings**

Informix IDS **technology has been reused and/or bundled with** *39* **other product offerings**

Cloudscape **technology has been reused and/or bundled with** *253+* **other product offerings**

Community Source in IBM Facilitates Reuse



Key Features:

- Access Control
- Product builds, fixes and test drivers
- Discussion Forums
- Reference information
- Defect Reporting
- Feature Requests
- Code Storage and Version Control
- Project Management

1,208 **active projects and** *26,149* **registered users**

Benefits

- **Reuse over reinvention**
- **Improving information flow**
- **Leveraging broader IBM**
- **Improving quality through peer reviews and user feedback**
- **Deliver more function on shorter schedule**
- **Most valuable assets get the most attention**
- **Facilitate development**

Business & Software Development Process

IPD – Integrated Product Development **process**

- **Structured, end-to-end process for** *managing business investment decisions and development efforts*
- **Methodology for defining, developing, qualifying, delivering, and supporting offerings**
- **Business life-cycle model for any type of offering**

IRUP – IBM (Internal) Rational Unified Process

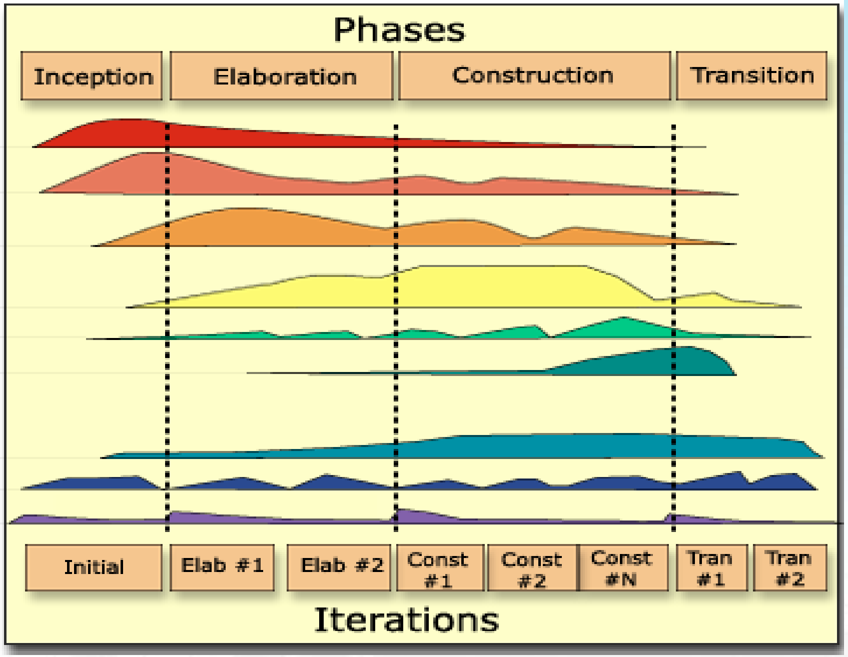
- **Customization of the Rational Unified Process to address the specific** *needs of IBM internal software development*
- **Underlying set of philosophies and practices for successful software development**
- **Software development life-cycle model for the development of software**

IBM Using Rational Unified Process = IRUP

Content and Structure

Disciplines

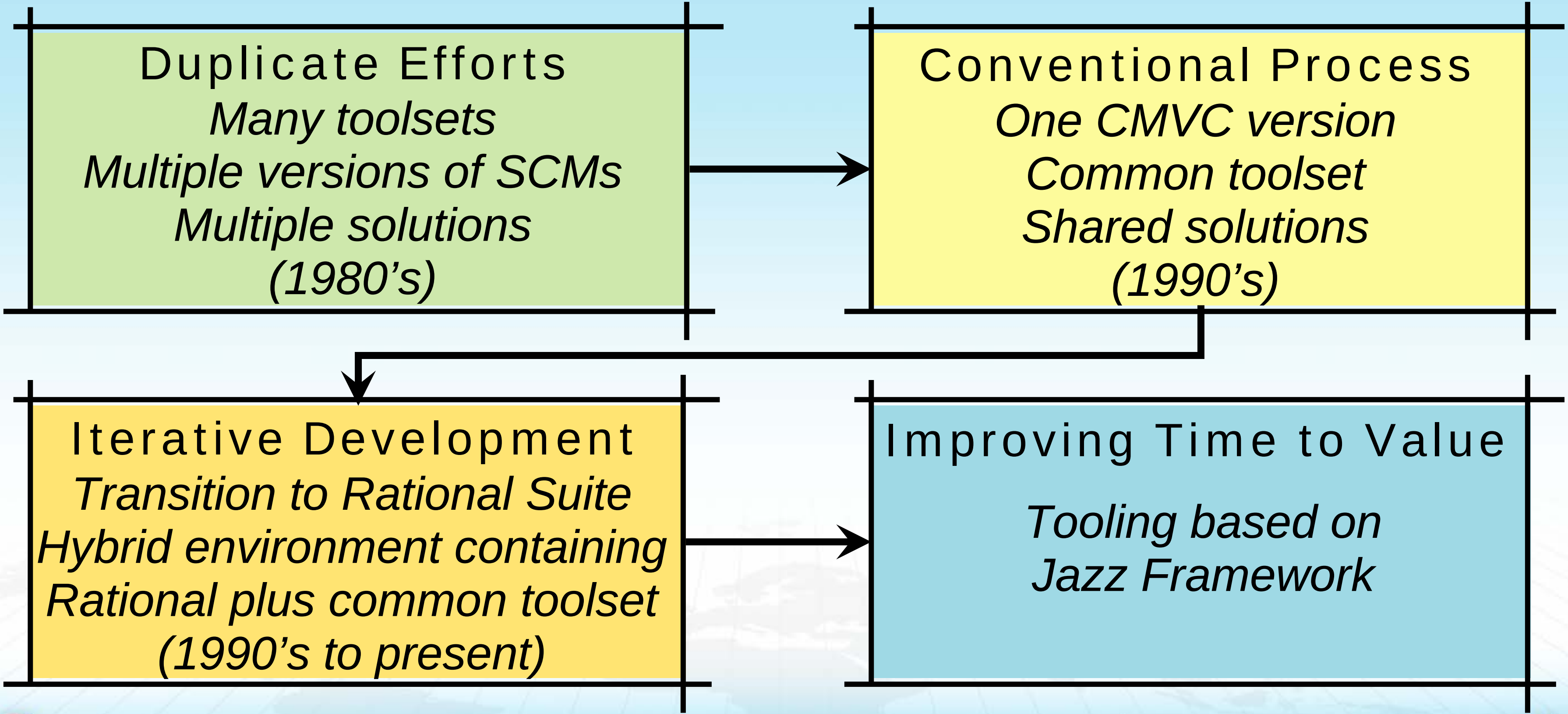
Business Modeling
Requirements
Analysis & Design
Implementation
Test
Deployment
Configuration and Change Management
Project Management
Environment



Time

IPD Phases: Concept | Plan | Develop | Qualify | Launch | Lifecycle

IBM Development Tool Evolution



Duplicate Efforts

Many toolsets
Multiple versions of SCMs
Multiple solutions
(1980's)

Conventional Process

One CMVC version
Common toolset
Shared solutions
(1990's)

Iterative Development

Transition to Rational Suite
Hybrid environment containing Rational plus common toolset
(1990's to present)

Improving Time to Value

Tooling based on Jazz Framework

Internal Adoption of Rational Solution

- Rational products deployed to the *"Right Team"* at the *"Right Time"*
- **Top adoption enablers**
  - Executive and technical commitment
  - Dedicated deployment engineers for Team Products (CC, CQ)
  - Support for internal IBM for ALL Rational tools
  - Rational Knowledge Sharing Center Web Site
    - Education – online training, forums
    - Best practices
    - Downloads
    - Beta support

Rational Software Development Team

Using Rational Tools for Rational Development

- Rational Unified Process
- Rational Asset Manager
- Rational ClearCase
- Rational ClearQuest
- Rational Team Concert
- Rational RequisitePro

600 users

- Rational Software Architect
- Rational Build Forge
- Rational Manual Tester
- Rational Functional Tester
- Rational Performance Tester

Used by 60+ Rational development efforts

DB2 V9 Development

on schedule
highest quality

Challenge

- **Deliver high quality product, on schedule, leveraging collaborative development effort of**

1,000 developers, spanning 12 labs, in 8 countries

Solution

- Rational ClearCase **for configuration management**
- Rational ClearQuest **for change management**

deployed worldwide as a standardized platform for cross-site development

IBM Tivoli Rome Lab Increases Productivity

Challenge

- Increase productivity of development and testing teams
- Improve quality of products

Solution

- Rational Unified Process
- Rational Software Architect
- Rational Functional Tester
- Rational Method Composer

Benefit

- 30% increase in developer productivity
- Requirements and design defects cut in half
- Test productivity increased by 20%, while test coverage increased by 30%
- 20% of all functional testing automated

Overall, 200% ROI on first product release using Rational Tools

WebSphere Application Server Development

900+ Developers

- China
- Canada
- Germany
- India
- Israel
- Japan
- UK
- USA

4.1M+ New LOC

- **325 separate build images**
- **900+ builds per week**
- **200 automated build tests**
- **50,000 Java cert tests**
- **35,000 function tests**
- **1,100+ customer scenarios**

Using the Right Tools

- *Rational Unified Process* **for "use cases" best practices**
- **Build tools, using** *Rational Build Forge*
- *Rational Application Developer* **for code modeling and development tools**
- **Automated GUI testing with** *Rational Functional Tester*
- **Stress testing with** *Rational Performance Tester*

Tivoli Storage Manager Testing

reduced
test effort by 90%
from 5 days → 3 hrs

Challenge

- **Development cycle of 12 months for major release and quarterly for maintenance put significant strain on test team**
- **Needed to automate test process to reduce cycle time and human error**

Solution

- Rational Functional Tester **for automating test bucket**

quickly assess and baseline overall quality for new product builds and maintenance releases

Tivoli

Best Practices for Distributed Development Success

Sound Development Governance Principles

- Lightweight central governance mechanisms
- Development Steering Committee
- Architectural Board
- Culture of sharing and reuse
- Developer Web site
- Centralized development services

+

Enable for Success

- Tools, not Rules
- Community source
- Shared asset repository
- Best practices
- Common components
- Clearing House for dependency management

+

Execute Agile / Lean for Productivity

- Discipline, adaptive development approaches
- Continuous stakeholder feedback to understand changing needs
- Time-boxed iterations
- Eliminate waste, increase visibility

=

Guiding Principles for Software Development

- Architecture Blueprint
- Outside-in Development
- Agile / Lean approaches
- Modeling and Componentization
- Fostering Communities and sharing Best Practices

THANK YOU

RU READY TO SAVE THE DAY

IBM Rational Software Development Conference 2008

WHERE TEAMS ARE R-HEROES

IBM

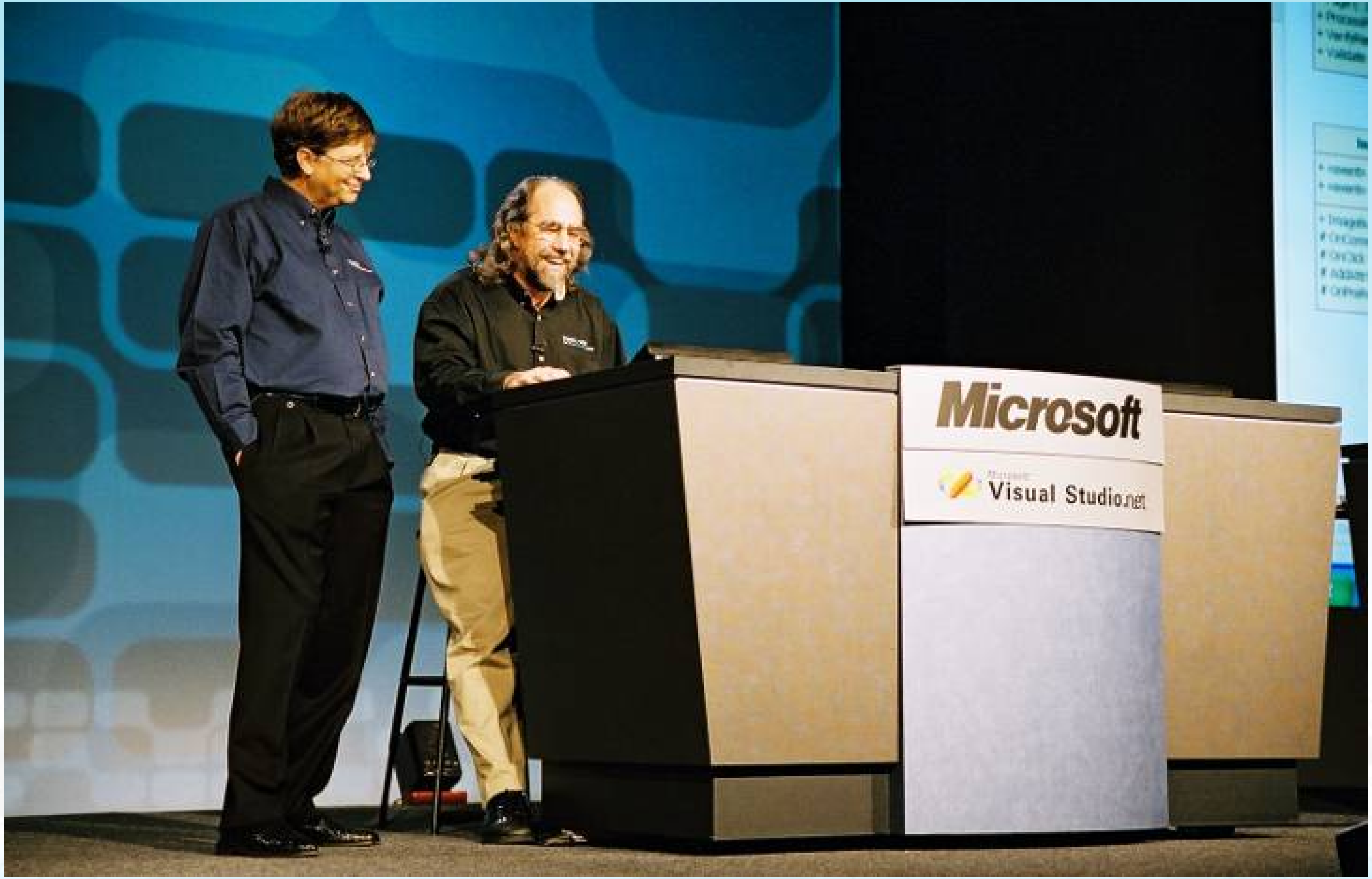
IBM Rational Software Development Conference 2008

WHERE TEAMS ARE R-HEROES

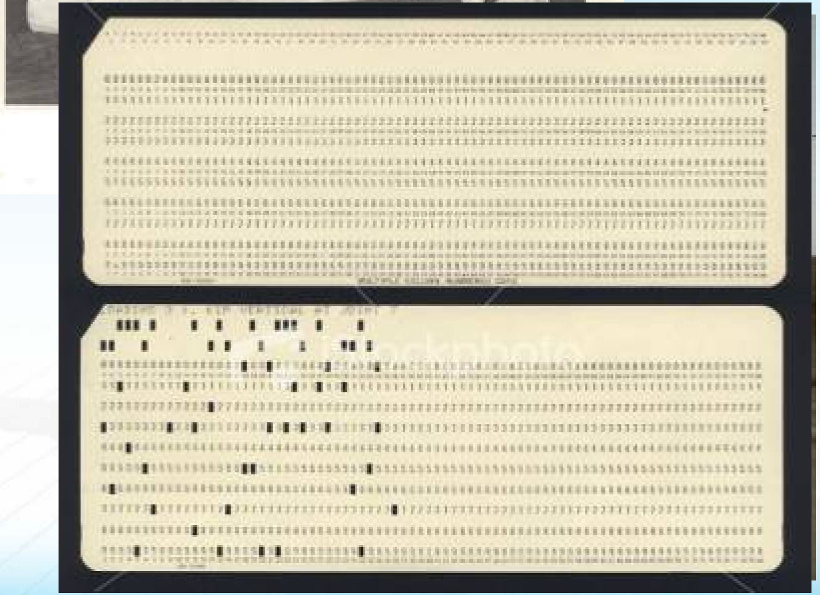
GRADY BOOCH

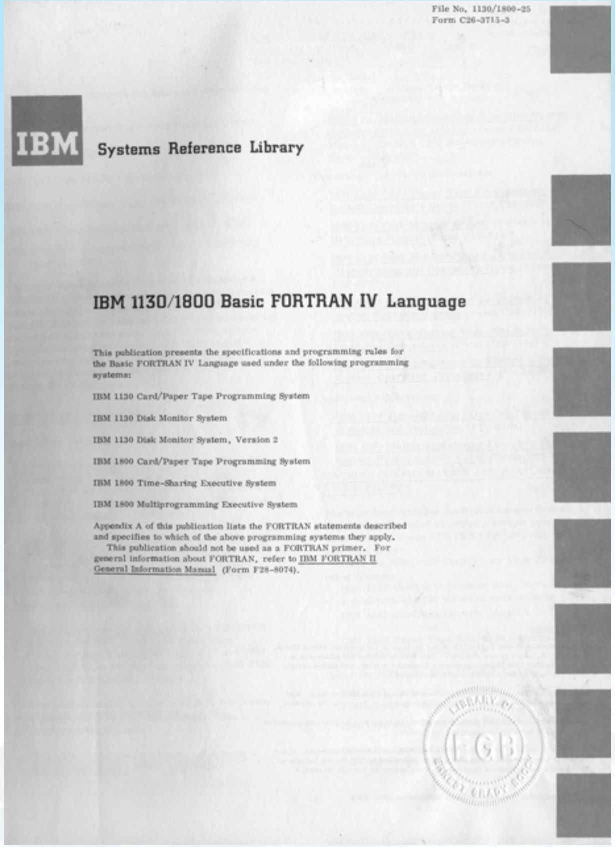
IBM FELLOW
CHIEF SCIENTIST FOR SOFTWARE ENGINEERING, IBM RESEARCH
FREE RADICAL

Rational. software


Microsoft
Microsoft
Visual Studio.net

The Future Of Software





File No. 1130/1800-25
Form C26-3715-3

IBM Systems Reference Library

IBM 1130/1800 Basic FORTRAN IV Language

This publication presents the specifications and programming rules for the Basic FORTRAN IV Language used under the following programming systems:

IBM 1130 Card/Paper Tape Programming System

IBM 1130 Disk Monitor System

IBM 1130 Disk Monitor System, Version 2

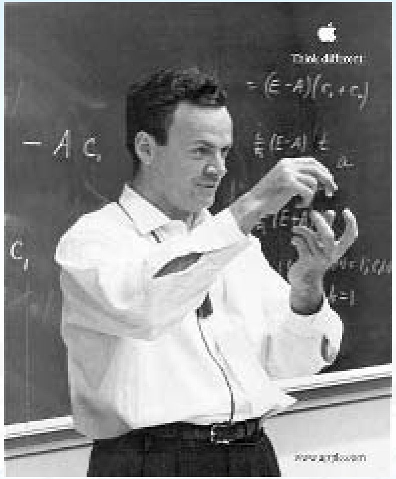
IBM 1800 Card/Paper Tape Programming System

IBM 1800 Time-Sharing Executive System

IBM 1800 Multiprogramming Executive System

Appendix A of this publication lists the FORTRAN statements described and specifies to which of the above programming systems they apply.

This publication should not be used as a FORTRAN primer. For general information about FORTRAN, refer to IBM FORTRAN II General Information Manual (Form F28-8074).


Think different.
www.apple.com

The Current State

The typical software-intensive system is

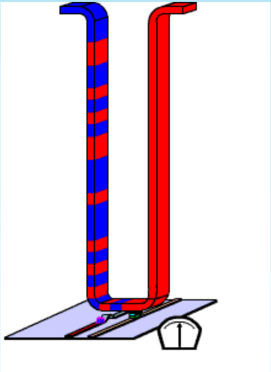
- Continuously evolving
- Connected, distributed, & concurrent
- Multilingual & multiplatform
- Secure & autonomic
- Developed by geographically- temporally-distributed teams

Most systems are actually systems of systems

- Services & other messaging mechanisms dominate
- Such systems encompass both hardware & software

Every advance leading
to the future state of
the world requires the
presence of software
yet-unwritten as of today

Growth Of Storage



- ***The production of data is growing***
  - Google processes 20 petabytes/day[1]
  - The Internet handles over 627 petabytes/day[2]
- ***Storage densities are increasing***
  - 200 gigabytes/inch^2 are common today
  - Racetrack memory could increase storage density by a factor of two (20,000 gigabytes/inch^2)[3]

[1] http://www.niallkennedy.com/blog/2008/01/google-mapreduce-stats.html

[2] http://en.wikipedia.org/wiki/Petabyte

[3] http://www.almaden.ibm.com/spinaps/research/sd/?racetrack

Growth Of Computational Power

- ***Computational power is abundant***
  - **A single BladeCenter can reach 7 teraflops**
  - **IBM Road Runner may reach one petaflop**
  - **Hardware costs are around 20 cents/gigaflop; operating costs are approximately 3 watts/gigaflop[1]**
- ***The frequency scaling wars are ending***
  - **At 10 atoms/transistor, quantum effects & power dissipation become critical issues issues**
  - **Multicore processors are becoming the norm**

Opportunity

[1] http://en.wikipedia.org/wiki/FLOPS

Growth Of Connectivity

Bandwidth is increasing

- **Copper may reach 10 gigabytes/second**
- **Wireless networks are becoming pervasive**

- ***Out of 3.7 billion IPv4 addresses[1]***

China	***19.246 million***
US	***13.610 million***
Germany	***5.414 million***
Italy	***3.881 million***
Indonesia	***3.465 million***
Taiwan	***3.455 million***

Opportunity

[1] http://www.bgpexpert.com/addressespercountry.php

Given relatively unlimited storage, abundant computational power, & pervasive connectivity…

What will future software-intensive systems look like?

How will we develop, deploy, & evolve such systems?

What is the value proposition?

Growth Of Storage

- Searching & indexing problems grow exponentially
- What are the privacy implications of having your entire life recorded & nothing forgotten?[1]
- Will we enter a digital dark age?[2]

Challenge

[1] http://www.guardian.co.uk/science/2005/dec/28/research.highereducation

[2] http://www.rense.com/general38/escap.htm

Growth Of Computational Power

- Writing correct software for intimate concurrency is a wicked problem
- Data center energy costs are becoming a limiting factor
- There’s no lack of sloppy software and/or hard problems that eat cycles
  - XML
  - Ray tracing
  - NP complete problems

Growth Of Connectivity

- Connectivity is unevenly distributed
- Opportunities for security breaches abound
- Opportunities for offensive cyberwarfare are emerging[1]

Challenge

[1] http://www.afcyber.af.mil/

Furthermore…

- How can you trust/have confidence in a system of systems over which you have no control of its parts?
- What are the implications for the globalization of systems development, deployment, & evolution?
- What are the economic implications of the commoditization of hardware & software?
- How does one address the inertia of legacy systems?

Design “Flaws” In The Web

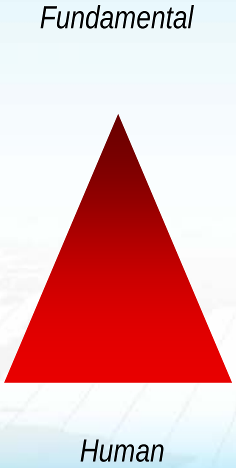
- **Poor separation of concerns between presentation & semantics**
  - **Evolution of the semantic web**
- **Address exhaustion**
  - **Moving from IPv4 to IPv6**
- **Changing assumptions regarding sessions**
  - **From stateless connections to always on video streaming**

Future Software-Intensive Systems

- **Future systems will be just like contemporary ones except they will be**
  - **More massive**
  - **More pervasive**
  - **More transparent**
  - **More critical**

Developing, Deploying, & Evolving

- **Limiting factors are rarely due to the laws of physics or the laws of software**
- **The wicked problem centers around the intrinsic human ability to manage complexity**



Laws of physics
Laws of software
Challenge of algorithms
Difficulty of distribution & concurrency
Problems of design
Importance of organization
Impact of economics
Influence of politics
Limits of human imagination

Value Proposition

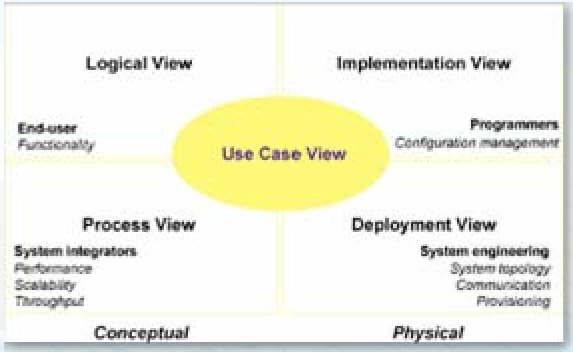
- **There are no limits to human imagination**
- **You can’t outsource innovation**

What We Know

- The fundamentals never go out of style
  - Craft crisp & resilient abstractions
  - Maintain a good separation of concerns
  - Create a balanced distribution of responsibilities
  - Refactor to simplicity
- Process
  - Grow a system through the incremental & iterative release of executable architectures

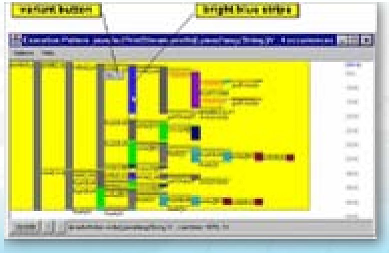
Software Architecture

- ***Every system has an architecture; most are accidental, some are intentional***
- ***Different stakeholders have different concerns & therefore different viewpoints***
- ***All well-structured software-intensive systems are full of patterns***



Multicore

- The average developer does not know how to build secure intimately concurrent software
- However, we can
  - Push some aspects to the operating system
  - Hide some complexity in compilers
  - Offer new programming languages & pattern languages
  - Provide better tools for debugging and visualization



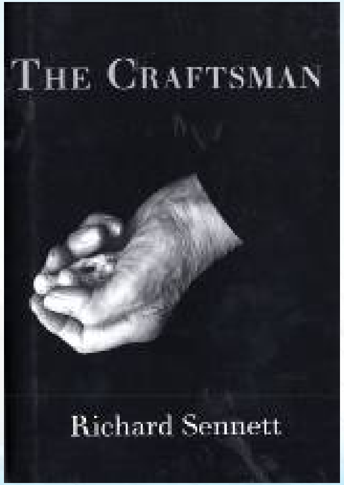
Collaboration

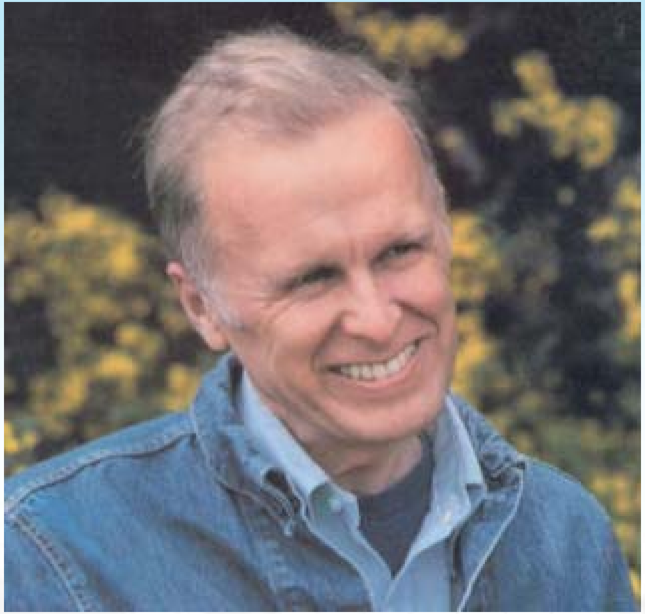
- Geographic distribution
  - Development across time zones and political boundaries
  - Issues of trust
  - The water cooler problem
- Temporal distribution
  - The preservation of tribal memory

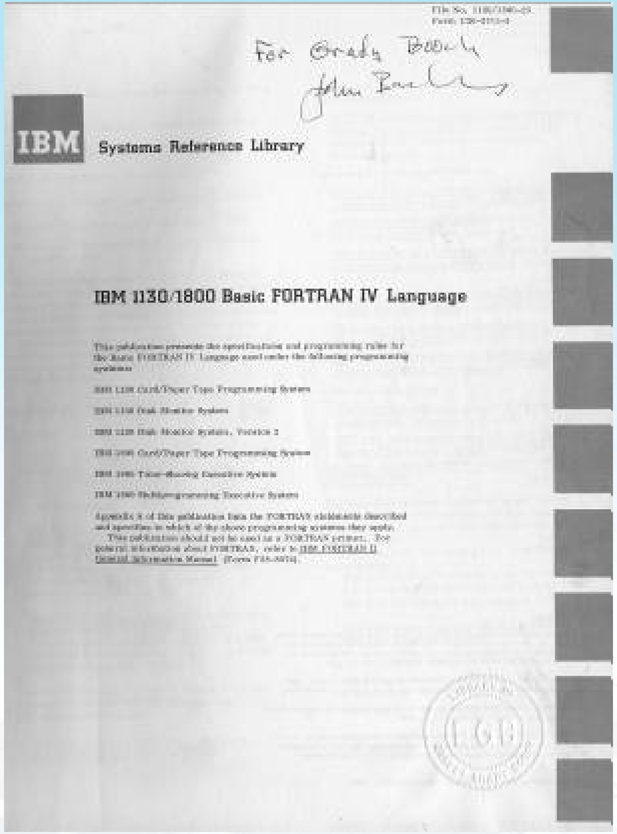
Every advance leading to the future state of the world requires the presence of software yet-unwritten as of today

Ethical/Moral/Legal Considerations

"All craftsmanship is founded on skill developed to a high degree….At its higher reaches, technique is no longer a mechanical activity; people can feel fully and think deeply what they are doing once they do it well. It is at the level of mastery… that ethical problems of craft appear."







File No. 1130/1800-25
Form C26-3715-4

For Grady Booch
John Backus

IBM Systems Reference Library

IBM 1130/1800 Basic FORTRAN IV Language

This publication presents the specifications and programming rules for the Basic FORTRAN IV Language used under the following programming systems:

IBM 1130 Card/Paper Tape Programming System

IBM 1130 Disk Monitor System

IBM 1130 Disk Monitor System, Version 2

IBM 1800 Card/Paper Tape Programming System

IBM 1800 Time-Sharing Executive System

IBM 1800 Multiprogramming Executive System

Appendix A of this publication lists the FORTRAN statements described and specifies in which of the above programming systems they apply.

This publication should not be used as a FORTRAN primer. For general information about FORTRAN, refer to IBM FORTRAN II General Information Manual (Form F28-8074).

Go be a hero to someone

IBM®

RU READY TO SAVE THE DAY

IBM Rational Software
Development Conference
2008

WHERE TEAMS ARE R-HEROES

On Today's Docket…

- **More Breakout Sessions**
- **Hands-on Workshops**
- **Birds-of-a-feather Sessions**
- **Lunch *(seating by product interest)***
  12:00 p.m. – 1:30 p.m.
- **Exhibit Hall & Solution Center Reception**
  12:00 p.m. – 2:00 p.m.; 4:30 p.m. – 8:00 p.m.
- **Spotlight Theater Presentations**
- **Jazz Café**
  9:45 a.m. – 11:00 a.m.; 2:00 p.m. – 4:00 p.m.
  Europe 3
- **Jazz Live! Poster Session & Reception**
  6:00 p.m. – 8:00 p.m.
  Pacific Hall C

jazz

Don't Forget!

Get your game card stamps for the Grand prize Giveaway

Don’t Miss Tomorrow’s Keynote!

8:00 a.m. – 9:30 a.m.

RU Ready to Boldly Go?

William Shatner
Actor, Author,
Philanthropist,
and Pop Culture Hero

Downloaded 100 sessions...

Powered by

EGL

Make sure to visit session
[illegible], Tuesday, 11:30am

Don’t Miss RSDC 2009 with your NEW Host:

Mitch Fatel

VP Marketing & Strategy

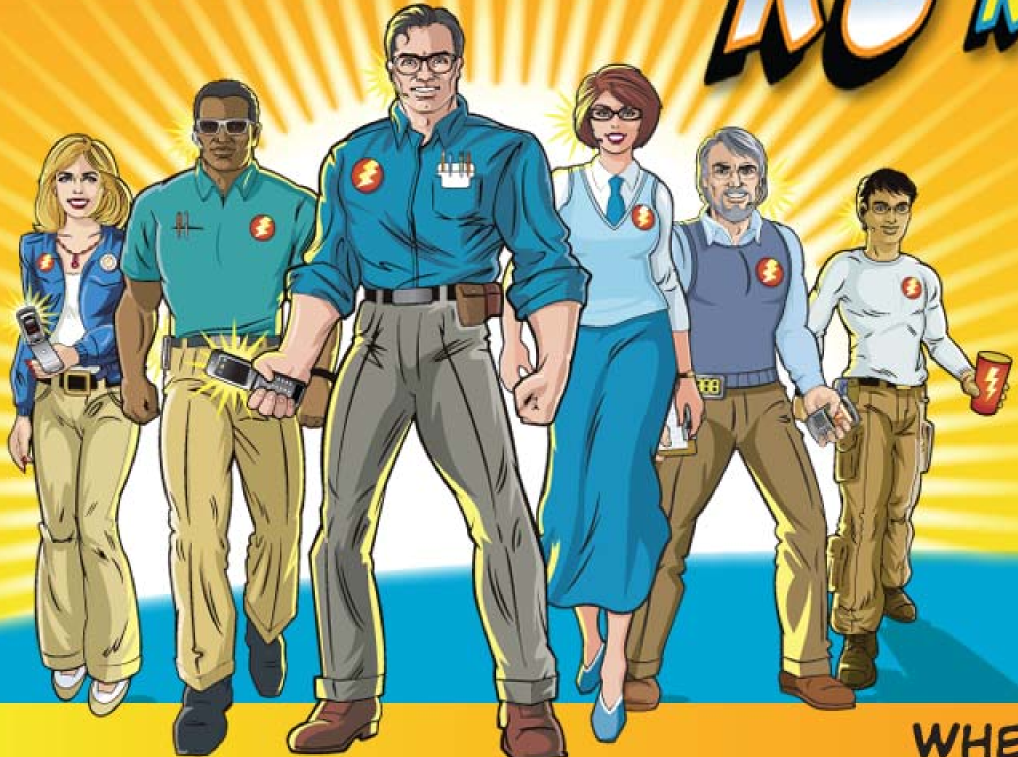
With ***possible*** guest speaker **….**

Scott Hebner

Title yet unknown

IBM®

RU READY TO SAVE THE DAY



IBM Rational Software Development Conference 2008

WHERE TEAMS ARE R-HEROES