4th International Conference on Service-Oriented Computing, 12/6/06

The Steady Path to Service-Oriented Computing

Dr. Alfred Z. Spector
azs@azs-service.com

December 6, 2006

Abstract

We have been climbing the path to Service-oriented computing since the dawn of the field of computer science. Contributing factors to our rate of progress have been the availability of hardware infrastructure, the availability of appropriate software infrastructure, and economic viability -- usually a function of the achievable economies of scale of re-usable services. Clearly our progress is also a function of the type (complexity) of service which is to be provided.

In this presentation, I will provide perspective on this service-oriented computing quest from a historical perspective, illustrate the very ambitious goals we should set for ourselves and their societal implications, and then highlight the great research and engineering problems we must solve if we are to continue to make great progress.

Outline

1. The Basics
   - The Broad Integration Agenda
   - The Computer Science Behind It.
   - The Forces Behind Service-Oriented Computing: Capability and Opportunity
2. Computational Architecture
3. Select Research Foci
   - Service-oriented Architectural Issues
   - Information Technology
   - Industry Services
4. Conclusions

Part 1: The Basics

There is a Broad Global Integration Agenda[1]

In fact, there are many:

- Broad societal goals
  - Trade
  - Education
  - Health care
- The firm or organization
  - Business-to-business; e.g., supply chain
  - Business-to-consumer; e.g., N:1, 1:N, …
- Science, engineering, education, military:
  - Collaborative innovation; multi-disciplinary R&D
- And many more

[1]From CMU 50th Anniversary lecture: http://www.cs50.cs.cmu.edu/UserFiles/File/SpectorSlides.pdf

Computer Science Behind the Integration Agenda

- **The primary foci of computer science are:**
  I. <u>Algorithms and their data structures</u>
    - Various methodologies, tools, and infrastructures that aid in understanding, specifying, programming, debugging, executing, verifying algorithms.
    - Creating new algorithms and data structures

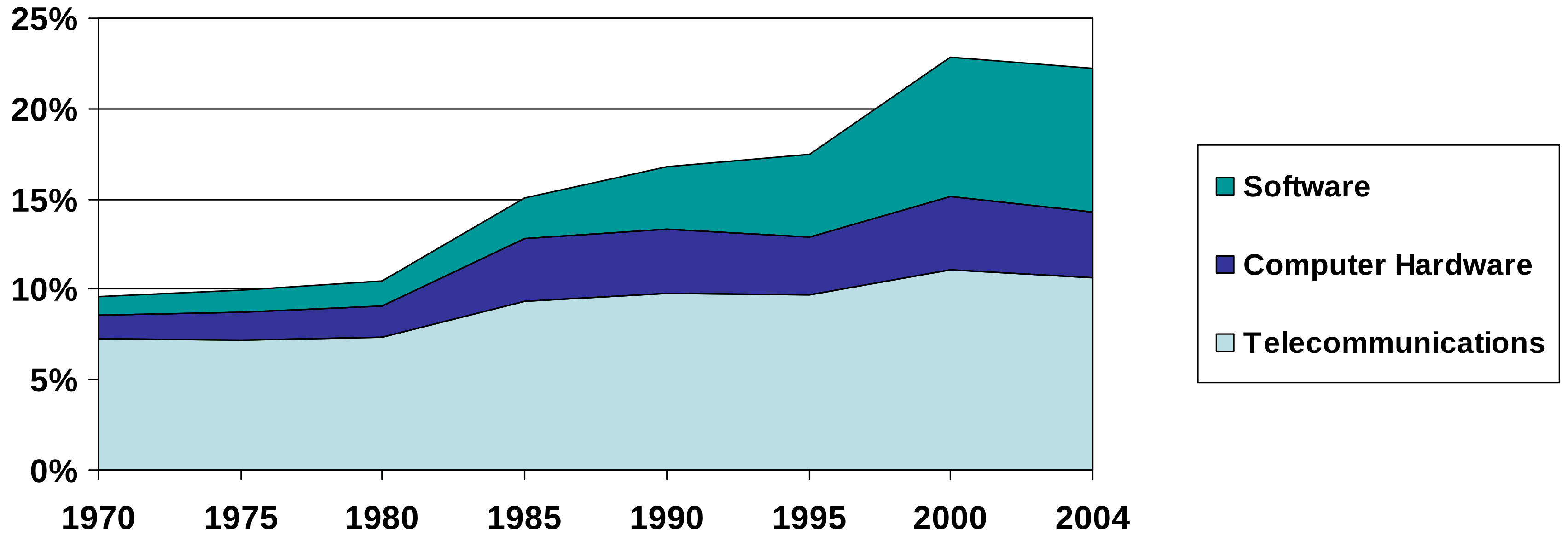
  II. <u>Composition of modules</u>.
    - Various methodologies, tools, and infrastructures for supporting understanding, specifying, programming, debugging, executing, verifying compositions created over time and space.
    - Creating architecturally valid components and compositions.

  The field's sub-disciplines often impact both I and II.
- **Clearly, key to both is abstraction, encapsulation, and re-use**
- **Of course, I and II overlap:**
  - An assembler program <u>composes</u> instructions
  - A mash-up <u>composes</u> rather bigger building blocks
- **<u>Nonetheless, the field has been steadily shifting more of its focus to the creation of assembly of larger, more flexible, abstractions.</u>**

Forces Behind Ascendancy of Composition

- **We can:**
  - Networking is ubiquitous
  - Computers are fast: 10^{15} increase in perf./cost of computation since 1900
  - Useful software methodologies, tools, and runtimes are manifold
- **We must:**
  - People, organizations, and their computations are distributed
  - Also, most useful programs are developed by many across time and space
- **We accept:**
  - Networked systems (including specifically the connection of one's own to others) are judged a part of life & business, with concomitant pros and cons
- **We want:**
  - Integration of information, process, and people
  - Integrated computation across functions, locations, organizations, etc
- **We need:**
  - Cost-effectiveness development derived from re-use
  - Efficient institutions using principles of comparative advantage & integration

There Are More Software Components to Integrate and Assemble



Source: Table 2.1, Current-Cost Net Stock of Private, Commercial Fixed Assets, Equipment and Software, National Bureau of Economic Analysis

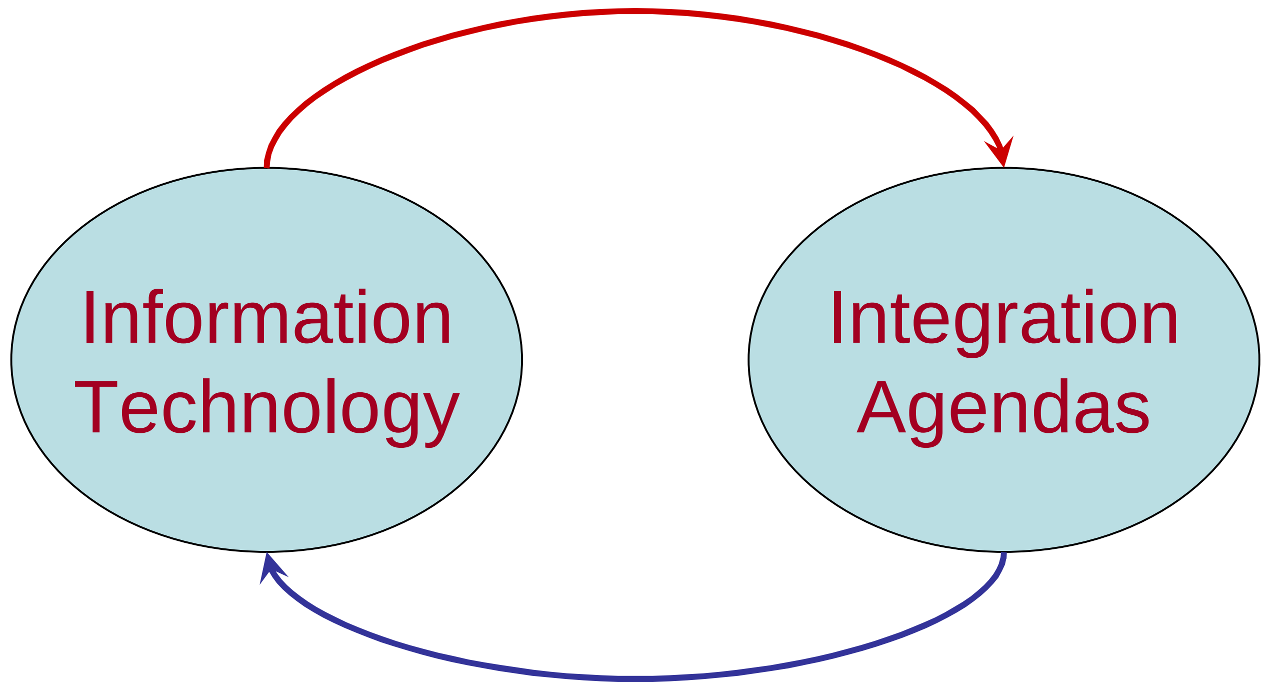
- I/T (particularly S/W) a Growing Share of U.S. Commercial Capital Stock[1]
- Note: Global I/T expenditure per year is roughly $1.5 x 10^12
- My analysis: Growth in I/T value is probably much steeper; why?
    - In terms of value, I/T has grown much faster due to price deflation
    - Software depreciation/expensing is demonstrably fast

[1]and by implication, the World's.

The Opportunity

- **Vastly increased capability of people and institutions**
  - Pooling of the world’s information technology capital stock creating unimaginably creative applications
    - Global integration of processes
    - Global integration of information
    - Global integration of people
  - This is a significant change in the utility of computation
  - This iceberg shows its tip in many ways:
    - Globalization
    - Vast changes, as IBM observes, in how services will be provided
    - Changes in retailing
    - Efficiency enhancements in the economy
  - Applicability to every organization and society
- **Decreasing costs for constant capability**
  - Re-use is a great idea for eventually, marginal cost of n+1st user tends to 0
  - Enforced standardization of services reduces churn

Information Technology & the Global Integration Agenda



- **Global I/T agenda drive demands for Information Technology**
- **Information Technology capabilities catalyze integration elsewhere**

Part 2: Computational Architecture

Computational Architecture

- **So, we accept abstraction, encapsulation, and re-use**
- **Thus, is "Service-oriented computing is a done deal?"**
- **Depends on what you mean**
  - If invocable abstraction is all, "yes, it's a done deal."
    - We've been pursuing it for years
    - E.g., RPC, threads, authentication, name service, transactions, etc…
  - But, the beauty is in the details:
- **So, it's now a given:**
  - There is computation and data available from many points in a network
  - There will be many defined functional entry points and invocations of these
- **But there are vastly more questions remaining as to the right and proper characteristics and engineering of service-oriented computing**
  - What are the complete semantics of the services?
  - What are the architectural structure of the world's core I/T and application services?
  - How does this ever growing architectural structure progress in time?
  - In implementation, how are services implemented and are they fairly static and centralized or adaptive and highly distributed in nature?

Spector’s Definition of Service-Oriented Computing[1]

Computing where can one create, flexibly deploy, manage, meter & charge for, secure, locate, use, and modify computer programs that define and implement well-specified functions having some general utility (Services), often recursively using other services developed and deployed across time and space, and where computing solutions can be built with a heavy reliance on use of these Services

Some key words used carefully within above paragraph:

Define and implement well-specified functions
Of general utility
Create
Flexibly deploy
Manage
Meter and charge for
Secure
Locate
Use
Modify
Recursive
Developed and deployed across time and space
Heavy reliance on use of these Services

[1]I think that Service-Oriented Computing is mostly synonymous with Service-oriented architecture (SOA).

Steady Path – A Long History

- Sabre ~1960
- Arpanet ~1970
- Internet ~1980
- Web ~1990[1]
- Web/Network Services (generic)? ~2000

[1] 1991 is more accurate, but less artistic

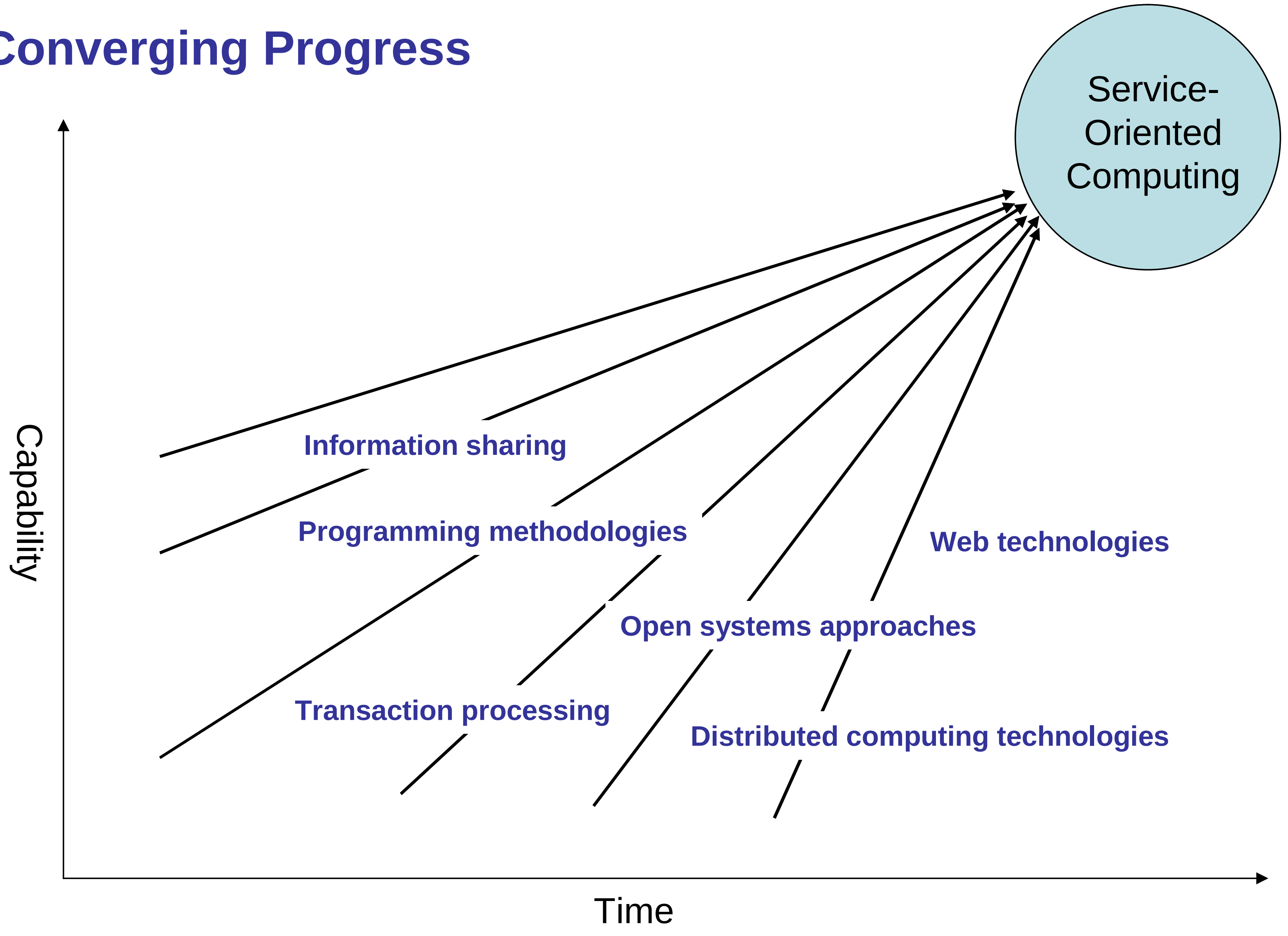
The Steady Path in More Detail

Many technologies converged over decades to support service-oriented computing

An incomplete list:

- **70’s and 80’s:**
  - Foundations of programming theory relating to OO - software engineering & languages
  - Foundations of distributed computing: Data sharing, function shipping
  - Runtimes and infrastructure: Threading, name-service, 3rd party authentication, authorization models, replication, caching, distributed transactions, queuing systems, email, ...
  - Sophisticated communication and invocation technologies
  - Data sharing architectures leading to SQL, distributed file systems, …
- **90’s**
  - Convergence into many systems of differential practicality: DCE, CORBA, DCOM , various Java initiatives, HTTP/HTML, XML, etc.
- **00’s**
  - Web Services
  - Growth in useful services and commercial exploitation
  - Increased research on the hard problems
  - (And more to come)

Converging Progress



The Missing Pieces:

- **Despite the efforts, there are missing pieces**
- **A subset of technical challenges at various times:**
  - Runtime
    - Insufficient transparency: location- and semantic-transparency, etc.
    - Quality of Service (QoS) issues
    - Availability
    - Performance penalties
    - Scalability problems
  - Security
    - Non-existent or weak security
    - Privacy issues
  - Programming complexity
  - Manageability and Cost-of-ownership
  - Insufficient universality and lack of standards
  - Economic models:Costing/Charging
  - And more

Many Attempts to Resolve I/T Technical Problems

- **There has been vast amount of effort to resolve technical problems**
- **Some of ones with which I have been personally involved:**
  - Semantically rich, efficient RPC, including Transactional RPC (TRPC)
  - Automated support for persistence: Camelot
  - Orthodox standards-based environments: Distributed Computing Environment (DCE)
  - Integrated authentication and ACL
  - Feature-rich runtimes: Encina Transaction Processing System
  - Consistent Relational, File, and Queued storage
  - …
- **The problems are hard & in many cases we still don’t have answers!**
  - Is a particular tech. capability necessary?
  - If so, exactly what should it be?

And Then, There is the Tragedy of the Commons

- **And then, there is the key problem for service creators illustrated by the Tragedy of the Commons:**
  - Setting:
    - There were 100 Cows on Boston Common
    - Each cow produced, say, 1 gallon of milk/day
    - Each owned by a family, F_i
    - But Family F_1 wanted more milk, and procured another cow
    - *Overgrazing* resulted in less milk production per cow, now only 0.98 gallon of milk/day
  - Results:
    - F_1 fairly happy getting almost 2 gallons per day
    - Society fairly unhappy, getting ~1% less total milk per day.
- **And so it often has been with reusable services**
  - A organizational subunit often builds its own service maximizing its own utility without regard to generality and cross-firm reuse
  - The result is a plethora of similar abstractions with high integration and maintenance costs
  - …at the expense of the broader firm

Part 3. Select Research Foci

- **Service-oriented Architectural and Engineering Research**
- **Service-oriented Systems Research**
- **Industry Services Research**

Service-oriented Architecture and Engineering Research

- **What methodologies?**
  - Key presumably is integration along many concerns
- **What programming tools?**
  - These must reflect the methodology
- **What management tools?**
  - These must consider the management of many cuts through the system: architecture, programming, runtime management, etc.
- **What economic models?**
  - Presumably many given breadth of applicability
- **How to manage complexity?**
  - (see next page)

Conclusions from The Conundrum of Systems[1]

- **Complexity grows despite all we have done in computer science, from Simon's *Sciences of the Artificial* to modern programming languages & software engineering techniques**
- **There is valuable, rewarding, and concrete work for Computer Science in combating complexity:**
  - Meaning
  - Measuring
  - Methodology
  - System Architecture
  - Science and Technology
  - Evolutionary Systems Design (recent addition)
  - Acknowledgment, Legal & Cultural Change
- **This area of work could prove as valuable as direct functional innovation: It requires focus**

[1]From my Dertouzos Lecture, MIT. http://www.csail.mit.edu/events/DLStalks/dlsspector03.html

Service-oriented Systems Research

- **How to you create robust systems without excessive programming complexity?**
  - E.g. When one must take service failure into account?
- **Basics**
  - Is there a, practical, normative general theory of consistency models? (e.g., whither atomicity?)
  - How are services implemented? Are they RPC invocation or a much more complex split between client and server?
  - How are security and privacy to be provided for the real world? Particularly, if you don't know what services being called?
  - How does one utilize parallelism, an increasingly important question in an era of lessening geometric clock-speed growth?
- **Management**
  - With an unbounded number of components, information hiding, etc., how can one manage systems?
  - How does one manage intellectual property?
- **Global properties**
  - Can one provide scalability generally?
  - How do we converge on universality and standards without bloat?
  - What systems can we deploy as really universal service repositories?
  - What technology can help solve the evolutionary system design problem?
- **Economics**
  - What are realistic costing/charging models and implementations
- **Social networking**
  - How do we apply social networking technology to help?

XML: A Critically Important Tool – Only a Beginning

- This is its 10th anniversary!
- Now, the predominant structure of electronic data interchange
- Multi-purpose markup language:
  - Program Specification and Implementation
  - Presentation Level formatting
  - Information Storage and Retrieval
- Adler, Cochrane, Morar, & Spector deem it a "Code of Integration"[1]
  - Consistency
  - Simplicity
  - Economy of scale
- A basis for working on semantic integration
- Vast economic, political, and social/cultural impact
- Most exploitation lies in the future: E.g., catalyze open source communities around the traditional business applications

[1] Technical context and cultural consequences of XML, S. Adler, B. Cochrane, J. Morar, & A.Spector, IBM Systems Journal, Vol. 45, No. 2 May 2006 **http://www.research.ibm.com/journal/sj/452/adler.html**

Industry Service Research

- **Need to have large (economy-wide) collection of business components**
  - Far more broad and complex than creation of I/T Standards
  - Fascinating challenge in evolution of businesses and economies
- **Creation and adoption of standards varies by industry:**
  - Standards efforts in some industries are immature and chaotic
  - If one standard is good, more are better ☹
  - Government regulation significant
- **The world is a big place**
  - E.g., China has a “do it yourself” culture. (Invent new rather than re-use)
  - Healthcare varies greatly by country
  - Vertical Industry organizations tend to be regional
- **XML is a critical enablers, but now we need the full exploitation**
- **Effective industry standards solve the Tragedy of Commons dilemma in a way that a firm cannot do itself**
- **All work must be done with evolutionary systems design in mind**

SOA Infrastructure Standards are Fairly Simple

W3C® WORLD WIDE WEB consortium

WS-I WEB SERVICES INTEROPERABILITY ORGANIZATION

OASIS

IETF

DMTF®

Java Java Community Process

eclipse

ISO International Organization for Standardization

A representative grouping of standards bodies relating to SOA

(Slide Courtesy of IBM)

Industry Standards Landscape is much richer (just examples)

Health Level Seven
Canadian Institute for Health Information
Institut canadien d'information sur la santé
AIAG
CENELEC
European Committee for Electrotechnical Standardization
Comité Européen de Normalisation Electrotechnique
Europäisches Komitee für elektrotechnische Normung
ETSI
ACORD
3GPP
A GLOBAL INITIATIVE
oma
Open Mobile Alliance
GS1
ARTS
Association for Retail Technology Standards
of the National Retail Federation
JavaPOS
Java for POS Devices
ITU
Unified Point of Service
UPOS
SWIFT
FRC
Financial Reporting Council
OLE POS
cen
European Committee for Standardization
Comité Européen de Normalisation
Europäisches Komitee für Normung
IHE
ismo
PCI INDUSTRIAL COMPUTER MANUFACTURERS GROUP
PICMG
XBRL
extensible Business Reporting Language
HUFO
NACHA
The Electronic Payments Association

Vast numbers of organizers and quantities/categories of participation

(Slide Courtesy of IBM)

Industry Services Opportunities and Challenges

- **Tremendous growth in standards needed to capitalize on large & growing information technology capital plant**

- **This will great variability across industry and around the globe**
  - Roles of industry group
  - Differential Regulation
  - Types Open Source
  - Interests of the nation-state

- **Partnerships between I/T and other industries needed**
  - Need combination of industry & I/T expertise; e.g., methodologies for
    - Creating usable standards
    - Working with competitors
    - Managing intellectual property

Conclusions

- **Computer Science will change the world far more in the future than it has in the past**
  - There are amazing computer science relating to service-oriented computing
  - There is very bold future research, interdisciplinary in nature, relating to both catalyzing and/or accommodating the integration agenda
  - I've characterized some of the research oppty's in the Architecture and Engineering, the Systems, and the Industry Services dimensions of Service-oriented computing.
  - The "virtual world" cliché is apt and we must help architect this properly.
- **Challenges about in architecture/software engineering, distributed systems, and focusing on industry standards.**
- **The multi-disciplinary, evolutionary nature of our field will create new types of computer scientists, new types of curricula, departments, and great new research breakthroughs**
  - My code for the indisciplinary research is: $\forall X_i$ (CS + X_i)

- **Toward our transformation to a Kinder, Gentler Borg! ☺**

Thank you very much!