

National Science Foundation Science of Design PI Meeting

Towards a Software Science of Design

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

Views herein are my own, and do not represent any other organization.

March 1, 2007

Towards a Software Science of Design

Abstract

(The process of) Design refers to the application of synthetic and analytic processes to plan and create new objects. Design is practiced across almost all disciplines. In software, design is practiced in many ways, not just the most classic C.S. approaches, but approaches across the entire software lifecycle. Software design is undergoing change due to the growing quantity and importance of software, changes in the platforms on which software runs, and growth in the locations across which software is built. Efficient and quality design are of growing importance to society and indeed to national welfare.

Similar software projects vary greatly in the results they achieve, implying there may be better and worse design practices. The field, however, has not been able to codify, systemize, and rapidly improve on best practices. In my opinion, a software science of design, integrative in nature, should be aimed at better understanding and codifying what we believe to be true, but also at creating better approaches and tools – all with the goal of facilitating software production across its lifecycle. This presentation includes an illustrative collection of relevant research and education topics, both basic and applied, that are core to a coherent, software science of design.

Outline

1. Design, Broadly
2. Software Design, Broadly
3. Changing Nature and Impact of Software
4. Successes, but Great Limitations
5. A Software Science of Design
6. Elements of a Research Agenda
7. Observations and Conclusions

Design, Broadly

- **The process of design refers to the application of synthetic and analytic processes to plan and create new objects.**
- **My perspective – great commonality in design across many human endeavors**
 - Protocols
 - (Medical) Differential Diagnoses
 - Bridge/Building Design
 - Proofs
 - Plays, Music, Novels
 - Urban Areas
 - Political Systems
 - Complex Systems
- **While design is central, it is not today a holistic discipline, studied in the same ways as the classical humanities or even science/engineering. Should it be?**
- **Cross-fertilization to be expected:**
 - AASHTO specifications for requirements of highway bridges¹
 - Design workbenches
 - Project design
- **Software design is often a part of broader designs so design is often inextricably linked**

¹Bridge Design and Construction, *CACM* 29(4): 267-283, 4/86, A. Z. Spector and D.K. Gifford

Software Design, Broadly

- **Design applied to many parts of aspects of software:**
- **The traditional cut:**
 - Requirements
 - Specifications
 - Implementation
 - Lifecycle Methodology and Tools/Systems
- **Other cuts**
 - Human interface design
 - Function to be presented
 - Stylistic considerations
 - Interface modeling and testing
 - Security design
 - Real world process considerations
 - Risk calculations
 - Choice of approaches
 - Design for Reliability
 - Reconciling opposite approaches
 - Design based on Hierarchical decomposition
 - Macro design
 - Micro design
 - Design for ... (e.g., other “ities”)

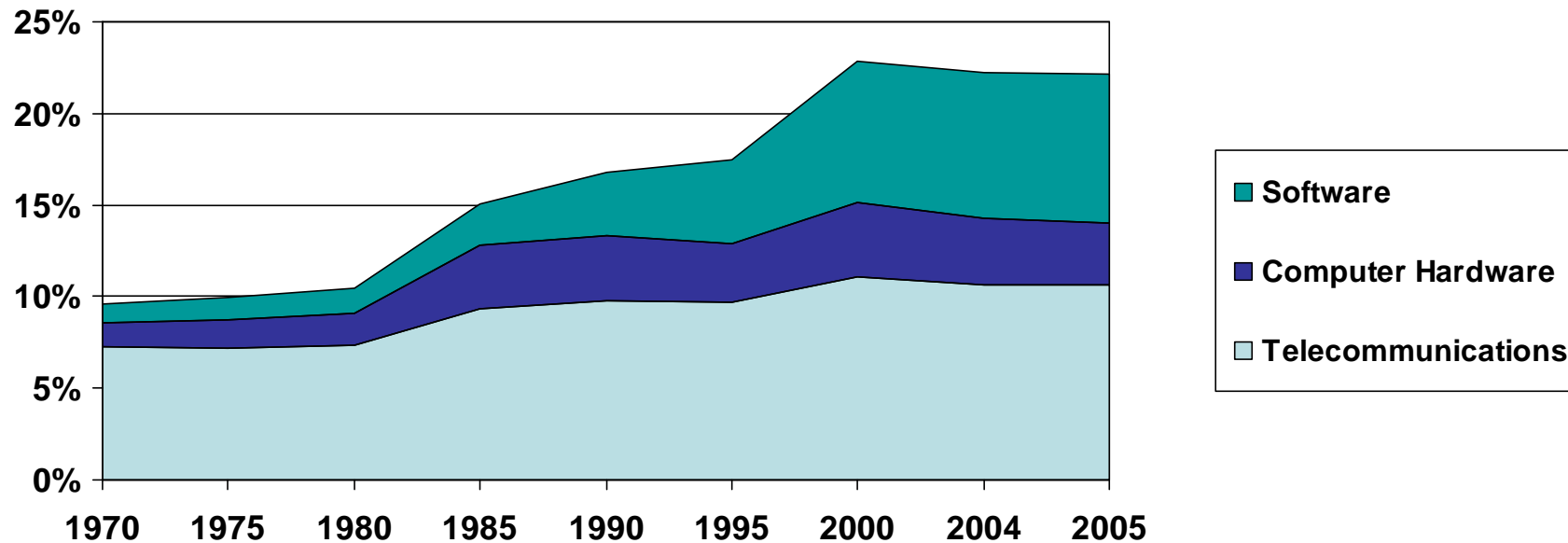
Difficulty of Software Design

- **Diversity of objectives and scale**
- **Quantity of artifacts**
- **Growing scale even for individual artifacts**
- **Post facto, dynamic assembly**
- **Malleability**
- **Longevity**
- **Absence of manufacturing cost**
- **Changing development possibilities**
- **Difficulty of verification**

Changing Nature of Software Design

- **Growing quantity:** *See chart on pg 8.*
- **Focus on composition & integration:** rather than low-level algorithms and small components *See slides on pg 9 & 10.*
- **Scale of software**
 - Size of modules
 - Number of modules
- **Platform changes:**
 - Volume,
 - Differential size
 - Location
 - Parallelism (coming)
- **Globalization of software development**
- **Growth in relative importance of software** *See slide on pg 11.*
 - Move towards more truly mission critical applications
 - Growth in aggregate importance of software inventory to society

There is More Software to Integrate and Assemble

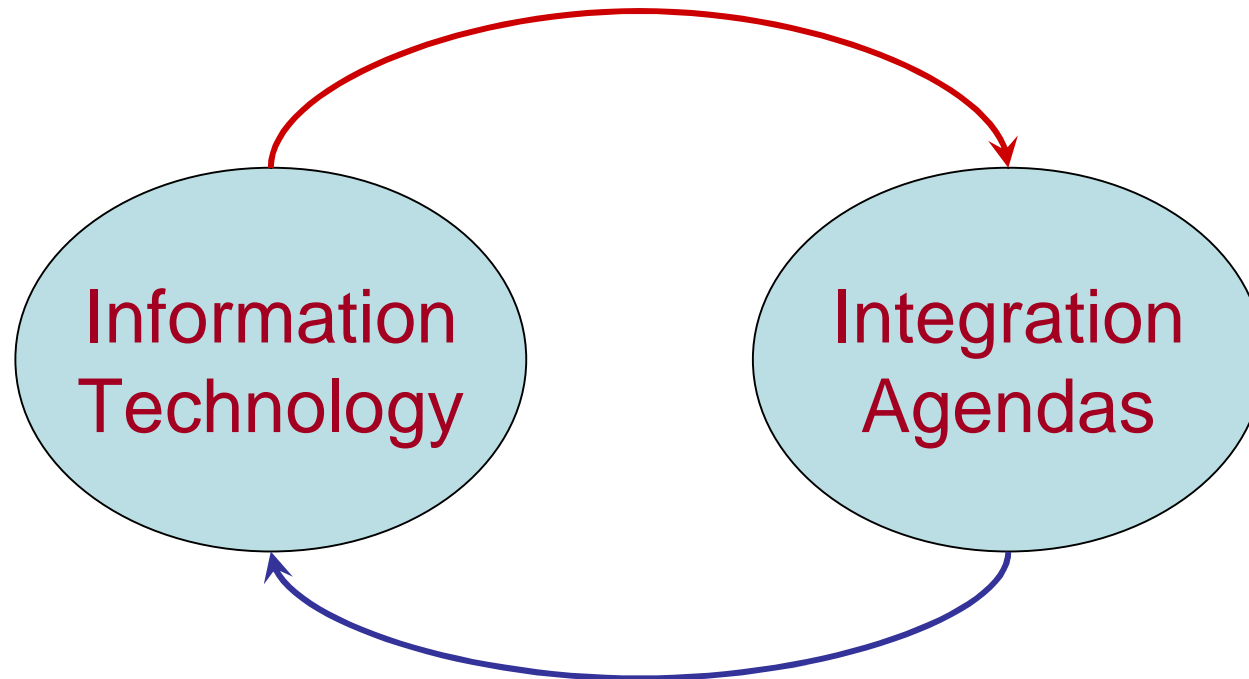


Source: Table 2.1, Current-Cost Net Stock of Private, Commercial Fixed Assets, Equipment and Software, National Bureau of Economic Analysis

- I/T (particularly S/W) a Growing Share of U.S. Commercial Capital Stock¹
- Note: Global I/T expenditure per year is roughly $\$1.5 \times 10^{12}$
- 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

¹and by implication, the World's.

Information Technology & the Global Integration Agenda¹



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

¹From my presentation The Steady Path to Services Oriented Computing @ Int'l Conf. on Service-Oriented Computing, Chicago, 2006 <http://conferences.ece.ubc.ca/icsoc2006/spector.pdf>

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 & 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

Other Thoughts on Software

- **Ever more important to efficient and proper operation of society**
- **Some truly great opportunities: health care, societal collaboration, empirical sciences, and they are being seized**
- **Software has the leverage to continue to allow great growth in productivity, productivity needed in the more competitive economy**
- **Software and information processing are inducing cultural change¹**

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>

Design of Software Systems

- **Clearly some successes**
 - High level languages
 - A growing number of low-level, re-usable components
 - A growing number of high-level, re-usable components (e.g., the mash-up, ...)
 - Certain methodologies
 - Much more...
- **However, huge problems persist**
 - Great variability in software project success
 - New projects and organizations typically redesign many aspects for the n th time
 - Cost, quality, vulnerability, usability, ... significant issues
 - The greatest organizations have significant problems
 - Commercial
 - Governmental
 - We have not been able to provide an engineering basis for software design; e.g.,
 - When to buy, build, innovate
 - How to approach security, quality
 - How to design project structures
 - How to manage intellectual property
 - How to evaluate quality of implementation, interface, etc.
 - The meaning of complexity (other than time or space)
- **That is, we do not really have a clear, reproducible basis for s/w development.**

Towards a Software Science of Design

A Software Science of Design would facilitate software production across the lifecycle by helping to understand, codify and integrate currently understood approaches to design, but also by creating better approaches and tools.

- A broad, integrative mission
- **Key foci:**
 - Cross-fertilization with other design disciplines
 - Holistic approach
 - Scientific analysis (not, “here’s a new approach, it’s real cool ☺”)
 - Education and standardization of approaches
 - Interdisciplinary work
- **Elements probably not included**
 - Domain-specific abstractions and their implementations
 - Elements directly covered in other areas of the field:
 - Programming Language Design
 - Verification
 - Etc.
 - These are admittedly blurry lines
- **Note:**
 - I don’t think that S/W design, broadly, will admit to as much standardization as exists in, say, road design
 - Nonetheless, there can still be families of approaches, parameterized (or meta-), approaches & tools...

Elements of a Research Agenda

- **Terminology and reference models**
- **Metrics**
- **Modularity and reuse**
- **Workbenches/platforms**
- **Development methodologies**
- **Automated assembly**
- **Design for change**
- **Design for security**
- **Complexity**
- **Education**
- **Team culture**
- **Impact of accounting and taxation**
- **Impact of liability, intellectual property, and anti-trust law**

Observations and Conclusions

- **The state of software design is not sound.** (1) We don't know enough (2) we rarely *codify* (in the broadest sense) what we know (3) we don't consistently use or educate what we codify.
- **Design is everywhere.** While particularly important in software, there should be a cross-discipline design community looking for synergies in innovation, education, and practice.
- **Design in software is multi-faceted.** It applies to all aspects of the lifecycle. A Science of Design should be broad enough to address all aspects.
- **Design is of growing importance.** Related not only to amount of software, more serious applications of software, greater diversity of it's development across time and space, and post-facto, dynamic assembly, with a greater base of components, design (but not programming) is oft. needed! Implications are manifold.
- **The research agenda has great opportunity.** There are plenty of topics for a broad research agenda. Work should proceed towards categorization and completeness: tops down and bottoms up.
- **Interdisciplinary research is of particular important.** The design of software is not purely a problem of computer science.
- **Establishment of objectives.** We should establish quantifiable objectives.

Thank you!

Backup Materials
(some details on research topics)

Research Topics 1

- **Terminology and reference models**
 - Where is design needed
 - What is the consistent terminology
 - What are the relationships between design disciplines
- **Metrics**
 - Productivity
 - Complexity
 - Quality
- **Modularity and reuse**
 - How to use modules from vast number of sources
 - Dealing with nesting issues that violate hierarchy
 - Approaches to successful re-use
 - Source code vs. black box, other artifacts
 - Patterns, objects, conversions, etc.
 - Metadata management
 - Specification of modules
 - Completeness
 - Labeling
 - Certification
 - Comparability

Research Topics 2

- **Workbenches/platforms**
 - Plenty of workbenches and platforms for design exist in certain subdomains
 - Integrated support across the lifecycle would seem to be needed
 - A rather complex topic given the scope, interconnectedness, and diverse source sof software
- **Development methodologies**
 - Specialization to task
 - The role of open source techniques
 - Extreme/waterfall/...
- **Automated assembly**
 - Can more design be automated?
 - Role of AI
 - Role of tagged semantics, semantic transformation

Research Topics 3

- **Design for change**
 - A topic essential and almost always done ad hoc
 - Principals for common use
 - Automated tools
 - Continuous update
 - Replication and change over
 - How to highlight changes
 - How to evaluate radical change opportunity
- **Design for security and robustness**
 - Almost always done ad hoc
- **Complexity grows despite all we have done in the field¹. Work in:**
 - Meaning
 - Measuring
 - Methodology
 - System Architecture
 - Science and Technology
 - Evolutionary Systems Design (recent addition)
 - Acknowledgment, Legal & Cultural Change

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

Research Topics 4

- **Education**
 - Should there be holistic software courses in design with topics including
 - Design via Integration agenda
 - Design for reuse
 - Teams & Collaboration
 - Lifecycle design
 - Should there be holistic courses in design across fields
 - Should there be degree programs led by topics in (higher-level) design?
- **Team culture**
 - Design of teams
 - Open source is doing a good job; why?
 - Impact of accounting and taxation
 - Certain financial rules may argue against proper expenditures
 - Economics of change
- **Impact of liability, intellectual property, and anti-trust law**
 - Impact of I.P. on reuse and sharing
 - Impact of anti-trust on integration and re-use/modularity
 - Impact of weak liability on quality, security, robustness