Progress toward an Engineering Discipline of Software

Mary Shaw
Institute for Software Research
Carnegie Mellon University
What does it mean to have an engineering discipline for software?

How far has software engineering progressed toward that goal?

What are the next steps?

with examples from civil engineering and software architecture
What is “engineering”?

Definitions abound

They have in common:

Creating cost-effective solutions ...
... to practical problems ...
... by applying scientific knowledge ...
... building things ...
... in the service of mankind

Engineering enables ordinary people to do things that formerly required virtuosos
What is “engineering”?

Definitions abound

They have in common:

Creating cost-effective solutions ...
... to practical problems ...
... by applying codified knowledge ...
... building things ...
... in the service of mankind

Engineering enables ordinary people to do things that formerly required virtuosos
Characteristics of engineering

- limited time, knowledge, and resources force decisions on tradeoffs
- best-codified knowledge, preferentially science, shapes design decisions
- reference materials make knowledge and experience available
- analysis of design predicts properties of implementation
Engineering evolves from craft and commerce; it requires scientific foundations, or at least systematically codified knowledge.

Exploiting technology requires both management and a body of codified knowledge.

Science often arises from progressive codification of practice.
Civil Engineering as Model
Civil Engineering

Example: Bridges and Arches
Great Buildings of the World
Bridges, Derrick Beckett,
Hamlyn Publishing Group, Ltd.,
London, England, pp 10, 12, 16, 19
Figure 4.4 Two Roman aqueducts, Anio Novus built on Claudia (From Curt Merckel, Die Ingenieurtechnik im Alterthum, 1899; courtesy Julius Springer-Verlag)
Craft of bridges

Romans
Renaissance & Industrial Revolution
Scientific Engineering

Empirical progress via failure and repair
No deliberate application of mathematics to determine size or shape
Little theory, but construction methods lasted until 19th century

Vitruvius: *De Architectura* [about 25 BC]
The Evolution of the Stone-arch Bridge
Fig. 28. Arch bridge, according to Leon Battista Alberti.

15th century
Ironbridge at Coalbrookdale, 1779
Dee Bridge disaster, 1847
Business of bridges

Romans

Renaissance & Industrial Revolution

Scientific Engineering

Increasingly long spans, lighter structures

Rules of thumb about proportions

Explanation of structures:
  - Brunelleschi on arches and domes 15th century
  - Galileo on beams 17th century

Introduction of cast iron, wrought iron, steel, and reinforced concrete
Hardest problem was identifying the proper basic concepts, e.g. force.

New mathematics was needed (calculus).
### Properties of Various Sections

<table>
<thead>
<tr>
<th>Section</th>
<th>Area of Section</th>
<th>Distance from Axis to Properties of Section</th>
<th>Moment of Inertia</th>
<th>Section Modulus</th>
<th>Radius of Gyration</th>
</tr>
</thead>
<tbody>
<tr>
<td>(bd - ab)</td>
<td>(y = \frac{d}{2})</td>
<td>(\frac{1}{12} (2bd^3 - 2ab))</td>
<td>(\frac{bd^2}{2} - d)</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
</tr>
<tr>
<td>(bd - ab)</td>
<td>(y = \frac{d}{2})</td>
<td>(\frac{1}{12} (2bd^3 - ab))</td>
<td>(\frac{bd^2}{2} - d)</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
</tr>
<tr>
<td>(bd - 2ab)</td>
<td>(y = \frac{d}{2})</td>
<td>(\frac{1}{12} (2bd^3 - ab))</td>
<td>(\frac{bd^2}{2} - d)</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
</tr>
<tr>
<td>(bd + bh)</td>
<td>(y = \frac{d}{2})</td>
<td>(\frac{1}{12} (2bd^3 + ab))</td>
<td>(\frac{bd^2}{2} + d)</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
</tr>
<tr>
<td>(bd + bh)</td>
<td>(y = \frac{d}{2})</td>
<td>(\frac{1}{12} (2bd^3 + ab))</td>
<td>(\frac{bd^2}{2} + d)</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
</tr>
<tr>
<td>(bd + 2ab)</td>
<td>(y = \frac{d}{2})</td>
<td>(\frac{1}{12} (2bd^3 - ab))</td>
<td>(\frac{bd^2}{2} - d)</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
</tr>
<tr>
<td>(bd + 2ab)</td>
<td>(y = \frac{d}{2})</td>
<td>(\frac{1}{12} (2bd^3 - ab))</td>
<td>(\frac{bd^2}{2} - d)</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
</tr>
<tr>
<td>(bd + 2ab)</td>
<td>(y = \frac{d}{2})</td>
<td>(\frac{1}{12} (2bd^3 - ab))</td>
<td>(\frac{bd^2}{2} - d)</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
</tr>
<tr>
<td>(bd + 2ab)</td>
<td>(y = \frac{d}{2})</td>
<td>(\frac{1}{12} (2bd^3 - ab))</td>
<td>(\frac{bd^2}{2} - d)</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
<td>(\sqrt{\frac{1}{\alpha}})</td>
</tr>
<tr>
<td>Sections</td>
<td>Area of Section $A$</td>
<td>Distance from Axis to Extremities of Section $y$ and $y_1$</td>
<td>Moment of Inertia $I$</td>
<td>Section Modulus $S = \frac{I}{y}$</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>---------------------</td>
<td>----------------------------------------------------------</td>
<td>----------------------</td>
<td>-----------------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$bd - ah$</td>
<td>$y = \frac{d}{2}$</td>
<td>$\frac{1}{12} (bd^3 - ah^3)$</td>
<td>$\frac{bd^3 - ah^3}{6d}$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$bd - ah$</td>
<td>$y = b - y_1$</td>
<td>$\frac{1}{3} (\frac{2mb^3 + ha^3}{2A} - Ay_1^2)$</td>
<td>$\frac{1}{y}$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$bd - 2ah$</td>
<td>$y = \frac{d}{2}$</td>
<td>$\frac{1}{12} (bd^3 - 2ah^3)$</td>
<td>$\frac{bd^3 - 2ah^3}{6d}$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$bd - 2ah$</td>
<td>$y = \frac{b}{2}$</td>
<td>$\frac{1}{12} (2mb^3 + ha^3)$</td>
<td>$\frac{2mb^3 + ha^3}{6b}$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$dm + mh$</td>
<td>$y = d - y_1$</td>
<td>$\frac{1}{3} \left( ry^2 + by_1^2 - 2a(y_1 - m)^3 \right)$</td>
<td>$\frac{1}{y}$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$dm + mh$</td>
<td>$y = \frac{b}{2}$</td>
<td>$\frac{1}{12} (mb^3 + ha^3)$</td>
<td>$\frac{mb^3 + ha^3}{6b}$</td>
<td></td>
</tr>
</tbody>
</table>
Figure 10.2 Types of arch bridge.

Figure 10.29 Coefficients for the design of a parabolic arch [59] \( M_p = C_1(EI/L^2) \).
Engineering of bridges

Romans

Renaissance & Industrial Revolution

Scientific Engineering

1700: good theories (statics, strength of materials)

1750: tabulations of properties of materials

1850: formal analysis of a bridge structure

1900: structural analysis worked out

1950: systematic theory

2000: design automaton
21st century

PennDOT now requires use of its software for automated design of simple bridges

- PennDOT’s Bridge Automated Design and Drafting Software (BRADD) automates bridge design from problem definition through CAD drawing.
- BRADD designs concrete, steel, and concrete bridges with spans of 18 feet to 200 feet.
Table 2.3-2 Matrix of Abutment Types versus Superstructure Types

<table>
<thead>
<tr>
<th>Superstructure Type</th>
<th>Abutment Type</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Traditional</td>
<td>Integral</td>
<td>SuperOnly High/Stub/Wall</td>
<td>SuperOnly Integral</td>
</tr>
<tr>
<td></td>
<td>High</td>
<td>Wall</td>
<td>Stub</td>
<td>High/Wall/Stub/Wall</td>
</tr>
<tr>
<td>Prestressed Concrete Adjacent Box Beam</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prestressed Concrete Spread Box Beam</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prestressed Concrete I-Beam</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Steel Rolled Beam</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Steel Plate Girder</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Evolution of civil engineering

1700: statics
1700: strength of materials
1775: hydraulics

1750: properties of materials
1850: full analysis of a bridge
Software Engineering
Software engineering as engineering

From the definition of engineering:

Creating cost-effective solutions ...
... to practical problems ...
... by applying codified knowledge ...
... building things ...
... in the service of mankind
Software engineering as engineering

From the definition of engineering:

The branch of computer science that ...

... creates cost-effective solutions ...

... to practical computing problems ...

... by applying codified knowledge ...

... developing software systems ...

... in the service of mankind

Software is design-intensive -- manufacturing costs are minor

Software is symbolic, abstract, and constrained more by intellectual complexity than by fundamental physical laws
"Software Engineering"

Rallying Cry

Phrase introduced 1968 to draw attention to “the software crisis”

Aspiration, not description

By some reports, “software engineering” was coined by Margaret Hamilton a few years earlier; the 1968 and 1969 NATO conferences brought the phrase into widespread use
Craft practice, 1968

- Monolithic development, merging research, development, production
- Software fine in many areas, but not for life-critical applications
- Widening gap between ambitions and achievement, increasing risk
- Software is late, over cost estimate, doesn’t meet specifications
- Too much revolution, not enough evolution

NATO Science Committee, 1968
Figure 2. From Selig: Documentation for service and users. Originally due to Constantine.
Production techniques

Systematic software development methods bring order and predictability to projects via structure and project management (1970-1990s)

- Structured programming
- Waterfall models
- Incremental and iterative development
- Cost/schedule estimation
- Process maturity
- Extreme, agile processes
Commerence drives science

Science is often stimulated by problems in commercial practice

- safety-critical tasks ➔ safety analysis
- large systems ➔ architectural patterns
- concurrency ➔ parallel logics & languages
- large state spaces ➔ model checking
- many versions ➔ program families, inheritance
- huge data sets ➔ MapReduce scalability
- adaptive systems ➔ MAPE model
Increasing Abstraction Scale

1950
- Macros
- Programming-any-which-way

1960
- Subroutines
- Separate compilation

1970
- Programming-in-the-small
- NATO SE conference
- Inheritance
- Abstract data types
- Information hiding

1980
- Programming-in-the-large
- Software development environments
- Object-oriented Patterns Packages Pipes and filters
- Inheritance
- Abstract data types
- Programming-in-the-large

1990
- Software architecture
- Model-driven development
- Component-based Systems
- Integrated product lines
- End user software engineering
- Web development tools
- Cloud computing architecture
- Service-oriented arch

2000
- Domain platforms
- Sociotechnical systems
- Vanishing system boundaries

2010
- Democratization of Internet

Beyond Programming
Fundamental ideas

**Abstraction** enables control of complexity

Imposing **structure** on problems makes them more tractable; **canonical solutions** exploit the structure

**Symbolic representations** are necessary and sufficient for solving information-based problems

Precise models support **analysis and prediction**

**Exponential growth** creates opportunities and limits
Design guidance

Choosing among solutions based on the problem setting

**Fig. 21.** Suggestions on the use of the various sorting techniques.
Software Architecture
Software architecture …

- ... is principled understanding of the large-scale structure of software systems as collections of elements that interact in distinct ways
- ... emerged 1990s from informal roots
- ... codifies a vocabulary for software system structures based on types of components and connectors
- ... provides guidance for explicit design choices bridging requirements to code
with a program transformation
A layered system!!

http://www.multicians.org/architecture.html
Craft practice

Software has always had structure

- Informal vocabulary
  - Objects, pipes/filters, interpreters, repositories...
- Intuitions and folklore about fitness to task

Ancient examples (since NATO69):

- Software bundled with hardware
- Compilers, layered operating systems
- Databases for accounting
FIGURE 7. Flight Computer Operating System (The FCOS dispatcher coordinates and controls all work performed by the on-board computers.)

<table>
<thead>
<tr>
<th>Client Layer*</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Access domain management</td>
<td></td>
</tr>
<tr>
<td>Buffering and record-level I/O</td>
<td></td>
</tr>
<tr>
<td>Transaction coordination</td>
<td></td>
</tr>
<tr>
<td>Agent Layer</td>
<td></td>
</tr>
<tr>
<td>Implementation of standard server interface</td>
<td></td>
</tr>
<tr>
<td>Logger, agent, and instance tasks</td>
<td></td>
</tr>
<tr>
<td>Helix Directories</td>
<td></td>
</tr>
<tr>
<td>Path name to FID mapping</td>
<td></td>
</tr>
<tr>
<td>Single-file (database) update by one task</td>
<td></td>
</tr>
<tr>
<td>Procedural interface for queries</td>
<td></td>
</tr>
<tr>
<td>Object (FID directory)</td>
<td></td>
</tr>
<tr>
<td>Identification and capability access (via FIDs)</td>
<td></td>
</tr>
<tr>
<td>FID to tree-root mapping; table of (FID,root,ref_count)</td>
<td></td>
</tr>
<tr>
<td>Existence and deletion (reference counts)</td>
<td></td>
</tr>
<tr>
<td>Concurrency control (file interlocking)</td>
<td></td>
</tr>
<tr>
<td>Secure Tree</td>
<td></td>
</tr>
<tr>
<td>Basic crash-resistant file structure</td>
<td></td>
</tr>
<tr>
<td>Conditional commit</td>
<td></td>
</tr>
<tr>
<td>Provision of secure array of blocks</td>
<td></td>
</tr>
<tr>
<td>System</td>
<td></td>
</tr>
<tr>
<td>Commit and restart authority</td>
<td></td>
</tr>
<tr>
<td>Disk space allocation</td>
<td></td>
</tr>
<tr>
<td>Commit domains</td>
<td></td>
</tr>
<tr>
<td>Cache</td>
<td></td>
</tr>
<tr>
<td>Caching and performance optimization</td>
<td></td>
</tr>
<tr>
<td>Commit support (flush)</td>
<td></td>
</tr>
<tr>
<td>Frame allocation (to domains)</td>
<td></td>
</tr>
<tr>
<td>Optional disk shadowing</td>
<td></td>
</tr>
<tr>
<td>Canonical Disk</td>
<td></td>
</tr>
<tr>
<td>Physical disk access</td>
<td></td>
</tr>
</tbody>
</table>

*Also called client Helix.

Figure 2. Abstraction layering.

A7E avionics architecture, as shown in Bachman et al
Software Documentation in Practice, SEI 2000
Commercial practice

1970s: batch processing
  o modules and procedure calls, Cobol

1980s: informal “architecture” in papers
  o colloquial use of architectural terms

1990s: early structure
  o software product lines

2000s: architecture research enters practice
  o company-specific overall architectures
  o frameworks, UML
  o objects everywhere
Commerce stimulates science

ad hoc structure, interoperability issues, design drift
multiple versions, variants, hardware
specialized application knowledge

styles /patterns → for software architecture
→ program families, inheritance
→ domain-specific models, languages
Sample idioms / styles / patterns

- **layers**
  - virtual machines <hierarchy of abstractions>
  - client-server systems <decomposition of function>

- **data flow**
  - batch sequential <indep. programs, batch data>
  - pipes and filters <transducers, data streams>

- **interacting processes**
  - communicating processes <processes, messages>
  - event systems <processes, implicit invocation>
## Architectural styles and reasoning

<table>
<thead>
<tr>
<th>Style class</th>
<th>Characteristic</th>
<th>Reasoning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data flow</td>
<td>Styles dominated by motion of data through the system, no “upstream” content control by recipient</td>
<td>Functional composition, latency</td>
</tr>
<tr>
<td>Closed loop control</td>
<td>Styles that adjust performance to achieve target</td>
<td>Control theory</td>
</tr>
<tr>
<td>Call-and-return</td>
<td>Styles dominated by order of computation, usually with single thread of control</td>
<td>Hierarchy (local reasoning)</td>
</tr>
<tr>
<td>Interacting processes</td>
<td>Styles dominated by communication patterns among independent, usually concurrent, processes</td>
<td>Nondeterminism</td>
</tr>
<tr>
<td>Data sharing styles</td>
<td>Styles dominated by direct sharing of data among components</td>
<td>Representation</td>
</tr>
<tr>
<td>Data-centered repositories</td>
<td>Styles dominated by a complex central data store, manipulated by independent computations</td>
<td>ACID properties, transaction rates, data integrity</td>
</tr>
<tr>
<td>Hierarchical</td>
<td>Styles dominated by reduced coupling, with resulting partition of the system into subsystems with limited interaction</td>
<td>Levels of service</td>
</tr>
</tbody>
</table>

Rules of thumb on data flow

If your problem is decomposed into sequential stages, consider *batch sequential* or *pipeline* architectures.

If each stage is incremental, so that later stages can begin before earlier stages finish, consider a *pipeline* architecture. But avoid if there is a lot of concurrent access to shared data.

If your problem involves transformations on continuous streams of data (or on very long streams), consider a *pipeline* architecture.

However, if your problem involves passing rich data representations, avoid pipelines restricted to ASCII.

Generality-power trades

Styles, Platforms, and Product Lines

Low

Data Flow
Call-Return
Events

Pipes & Filters
Process Control

Generic Styles

Power

High

Bosch Engine Control
Siemens Healthcare for 3D

Domain-Specific
Component Integration Platforms

Generic Component
Integration Platforms

AUTOSAR
HLA
IOS

Product Lines

Maturation of scientific ideas

Basic Research
Recognize problem, Invent ideas

Concept Formation
Refine ideas, publish solutions

Development & Extension
Try it out, clarify, refine

Internal Exploration
Stabilize, port, use for real problems

External Exploration
Broaden user group, extend

Popularization
Propagate through community

15-20 years

Sam Redwine, Jr. and William Riddle: Software Technology Maturation, Proc ICSE-8, May 1985
Maturation of software architecture

Foundations

1985-93: Basic Research
- Identify common idioms and styles, product lines, idea of connector

1992-96: Elaboration of models
- Architecture description languages, taxonomies, views, early formalization

1995-00: Unification, 2nd generation concepts
- Interoperability, integration, textbooks, conferences

1996-03: Explicit use in design, quality attributes
- Analysis, evaluation, formalized designs

1998-now: Tech useful in practice, UML, CBSE, company arch
- External Exp/Ext

2000-now: Commercialization, education
- Frameworks, web-fueled patterns
- Popularization

Garlan and Shaw. Software architecture: reflections on an evolving discipline. ESEC/FSE keynote 2011
Explanations for practitioners

N-Tier architecture

Virtual machine

http://www.codeproject.com/Articles/430014/N-Tier-Architecture-and-Tips

http://www.pcmag.com/encyclopedia/term/53927/virtual-machine
Systematically Organized Knowledge

SEI Series organizes knowledge about architecture and its analysis
Systematically Organized Knowledge

Pattern books for software architecture are emerging
AN x64 PROCESSOR IS SCREAMING ALONG AT BILLIONS OF CYCLES PER SECOND TO RUN THE XNU KERNEL, WHICH IS FRANTICALLY WORKING THROUGH ALL THE POSIX-SPECIFIED ABSTRACTION TO CREATE THE DARWIN SYSTEM UNDERLYING OS X, WHICH IN TURN IS STRAINING ITSELF TO RUN FIREFOX AND ITS GECKO RENDERER, WHICH CREATES A FLASH OBJECT WHICH RENDERS DOZENS OF VIDEO FRAMES EVERY SECOND BECAUSE I WANTED TO SEE A CAT JUMP INTO A BOX AND FALL OVER.

http://xkcd.com/676/
But is it “Engineering” yet?
But is it “Engineering” yet?

“Engineering” is associated with a level of assurance that protects the public health, safety, and welfare.
But is it “Engineering” yet?

“Engineering” is associated with a level of assurance that protects the public health, safety, and welfare.

Consider, though . . .

- Toyota unexpected acceleration, $1.6B payout
- 378 US data leaks in 2016, over 11M records
- Update bug destroys Hitoma x-ray satellite
- SWIFT (banking) network forged messages
- HBSC: 275K salary payments not processed
- Hackers remotely hijack a car (with permission, but …)
- . . .
Characteristics of engineering

- limited time, knowledge, and resources force decisions on tradeoffs
- best-codified knowledge, preferentially science, shapes design decisions
- reference materials make knowledge and experience available
- analysis of design predicts properties of implementation
software development methods

PRODUCTION

SCIENCE

COMMERCIAL

CRAFT

~ 1990, adoption of development methods

emerging, but spotty
Want to be part of this?

isri.cmu.edu/education/
isri.cmu.edu/jobs/tenure-track-se.html

Making Progress
Structural disruptions

indexing in edited content
programming
periodic releases
pure code
professional developer
trained users
Structural disruptions

indexing in edited content >> search
programming >> composition, evolution
periodic releases >> continuous update
pure code >> cyber-social adaptive systems
professional developer >> casual developer
trained users >> naïve users

These do not change the fundamental principles, but they change the challenges and the application of the principles.
Transmitting design knowledge

- Historical vehicles
  - word of mouth, rules of thumb
  - training in procedures
  - manuals
  - handbooks
  - textbooks and tutorials
  - standards
  - journals
  - tradeoff guidance
Transmitting design knowledge

- **Historical vehicles**
  - word of mouth, rules of thumb
  - training in procedures
  - **Manuals**
    - formerly, still some bricks
  - **handbooks**
    - publication cycle too slow
  - textbooks and tutorials
  - standards
    - relatively weak
  - journals
  - tradeoff guidance
    - largely missing

*How do we bring codified knowledge to design?*

*Exhortation won’t work*
Transmitting design knowledge

- Modern software engineering vehicles
  - tools that embody knowledge
  - frameworks and skeletons
  - design patterns
  - search in self-help forums like stackoverflow
  - search in code base (doesn’t help with design)

- Missing tools
  - proper documentation, specifications
  - guidance for choosing among designs
  - search in well-curated knowledge base
  - analog of MapReduce for software knowledge?
Architectures at scale

- Highly distributed, dynamically-formed task-specific coalitions of distributed autonomous resources (fix “mashups”)
- Agility, “perpetual beta”, live user testing (the cloud allows poor engineering practice)
- Pervasive cyber-physical systems: control, security, adaptation (“Internet of Things”)
- Socio-technical ecosystems: platforms, extensions, and people as part of system (“wicked” problems, end user development)
Scaling cost to consequence

Catastrophe

Greatest Need for Engineering Discipline

Consequences of Failure

Degree of Oversight

Convenience

Full oversight, manual operation

None: full automation, unattended operation

Therac-25
Drug interactions
Ambulance scheduling
Car cruise control
IRS 1040 on your own
Web search health info
Appointment scheduling
Finding restaurant
IRS 1040 TurboTax
Advance cruise control
Patient monitoring
Medical implants
Self driving car
Stocks program trading
Near real time weather
Automatic sports stats
Nuclear safety devices
Missile guidance

Scaling cost to consequence

Catastrophe

Consequences of Failure

Therac-25
Drugs interactions
Ambulance scheduling
Patient monitoring
Advanced cruise control
Self driving car
Nuclear safety devices

Greatest Need for Engineering Discipline

IRS 1040
IRS 1040 TurboTax
Web search health info
Appointment scheduling
Finding restaurant
Stock market alerts
Near real time weather
Automatic sports stats

Inconvenience

Consequences of Failure

Finding restaurant
Appointment scheduling
Web search health info
IRS 1040 on your own
IRS 1040 TurboTax
Stock market alerts
Near real time weather
Automatic sports stats

Degree of Oversight

Full oversight, manual operation
None: full automation, unattended operation

There are lots of casual developers

Estimated counts in American workplace

End

Users

Non-professional, non-managerial
47M

Professional (non-software), managerial
40.5M

Computer scientist, systems analyst
0.95M

Computer programmer
0.57M

Computer software engineer
0.98M

Self-taught
41.8%

BS in CS (or related)
37.7%

On-the-job training
36.7%

MS in CS (or related)
18.4%

Online class
17.8%

Some univ, no degree
16.7%

Industry certification
6.1%

Other
4.3%

Boot-camp
3.5%

PhD in CS (or related)
2.2%

Mentorship program
1.0%

“Professional and enthusiast programmers”
(international)


### Generations Online 2010

This chart shows the popularity of internet activities among internet users in each generation.

**Key:** % of internet users in each generation who engage in this online activity

<table>
<thead>
<tr>
<th>Activity</th>
<th>Millennials Ages 18-33</th>
<th>Gen X Ages 34-45</th>
<th>Younger Boomers Ages 46-55</th>
<th>Older Boomers Ages 56-64</th>
<th>Silent Generation Ages 65-73</th>
<th>G.I. Generation Age 74+</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Search</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Health info</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Social network sites</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Watch video</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Get news</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Travel website</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Travel reservations</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Buy a product</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Listen to music</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Social network sites</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bank online</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Online classifieds</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bank online</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Financial info</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Play games</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read blogs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Financial info</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rate things</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Religious info</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Online auction</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Podcasts</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Donate to charity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blog</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Virtual worlds</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Source:** Pew Internet surveys.

http://www.pewinternet.org/2010/12/16/generations-2010/
<table>
<thead>
<tr>
<th>Email</th>
<th>Search</th>
<th>Email</th>
<th>Search</th>
<th>Email</th>
<th>Search</th>
<th>Email</th>
<th>Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>Health info</td>
<td>Get news</td>
<td>Health info</td>
<td>Get news</td>
<td>Health info</td>
<td>Get news</td>
<td>Health info</td>
<td>Get news</td>
</tr>
<tr>
<td><strong>Social network sites</strong></td>
<td><strong>Govt website</strong></td>
<td><strong>Travel reservations</strong></td>
<td><strong>Buy a product</strong></td>
<td><strong>Travel reservations</strong></td>
<td><strong>Buy a product</strong></td>
<td><strong>Govt website</strong></td>
<td><strong>Watch video</strong></td>
</tr>
<tr>
<td>Get news</td>
<td>Watch video</td>
<td>Bank online</td>
<td>Social network sites</td>
<td>Bank online</td>
<td>Social network sites</td>
<td>Bank online</td>
<td>Social network sites</td>
</tr>
<tr>
<td>Buy a product</td>
<td>IM</td>
<td>Financial info</td>
<td>Social network sites</td>
<td>Financial info</td>
<td>Rate things</td>
<td>Religious info</td>
<td>Social network sites</td>
</tr>
<tr>
<td>Listen to music</td>
<td>Travel reservations</td>
<td>Online classifieds</td>
<td>Social network sites</td>
<td>Online classifieds</td>
<td>Online classifieds</td>
<td>Religious info</td>
<td>Social network sites</td>
</tr>
<tr>
<td>Online classifieds</td>
<td>Bank online</td>
<td>Financial info</td>
<td>Social network sites</td>
<td>Financial info</td>
<td>Rate things</td>
<td>Social network sites</td>
<td>Social network sites</td>
</tr>
<tr>
<td>Bank online</td>
<td>IM</td>
<td>Listen to music</td>
<td>Social network sites</td>
<td>IM</td>
<td>Listen to music</td>
<td>Financial info</td>
<td>Social network sites</td>
</tr>
<tr>
<td>Govt website</td>
<td>IM</td>
<td>Financial info</td>
<td>Social network sites</td>
<td>IM</td>
<td>Social network sites</td>
<td>Religious info</td>
<td>Social network sites</td>
</tr>
<tr>
<td>Play games</td>
<td>IM</td>
<td>Read blogs</td>
<td>Social network sites</td>
<td>IM</td>
<td>Religious info</td>
<td>Rate things</td>
<td>Social network sites</td>
</tr>
<tr>
<td>Read blogs</td>
<td>Financial info</td>
<td>Religious info</td>
<td>Rate things</td>
<td>Financial info</td>
<td>Religious info</td>
<td>Religious info</td>
<td>Rate things</td>
</tr>
<tr>
<td>Financial info</td>
<td>Religious info</td>
<td>Rate things</td>
<td>Rate things</td>
<td>Religious info</td>
<td>Religious info</td>
<td>Religious info</td>
<td>Religious info</td>
</tr>
</tbody>
</table>
Civilizing the electronic frontier

Civilization means

- Infrastructure and amenities
- Civil order, shared obligations, rule of law
- Empowering citizens to manage their own affairs
- Clarity on personal security/responsibility
- Product quality warranty and liability
This requires

- Policy informed by technology...
- ...balancing anonymity and responsibility
- ...balancing corporate and individual goals
- Implementation informed by societal needs...
- ...accepting the nature of “wicked problems”
- Widespread understanding of technology...
- ...and shared expectations about its use
- ...and usable user models for systems
Recapitulation

Engineering evolves from craft and commercial practice via science

Ideas evolve over time from pure research to practical production

The greatest need for engineering is in the most critical applications