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 ...

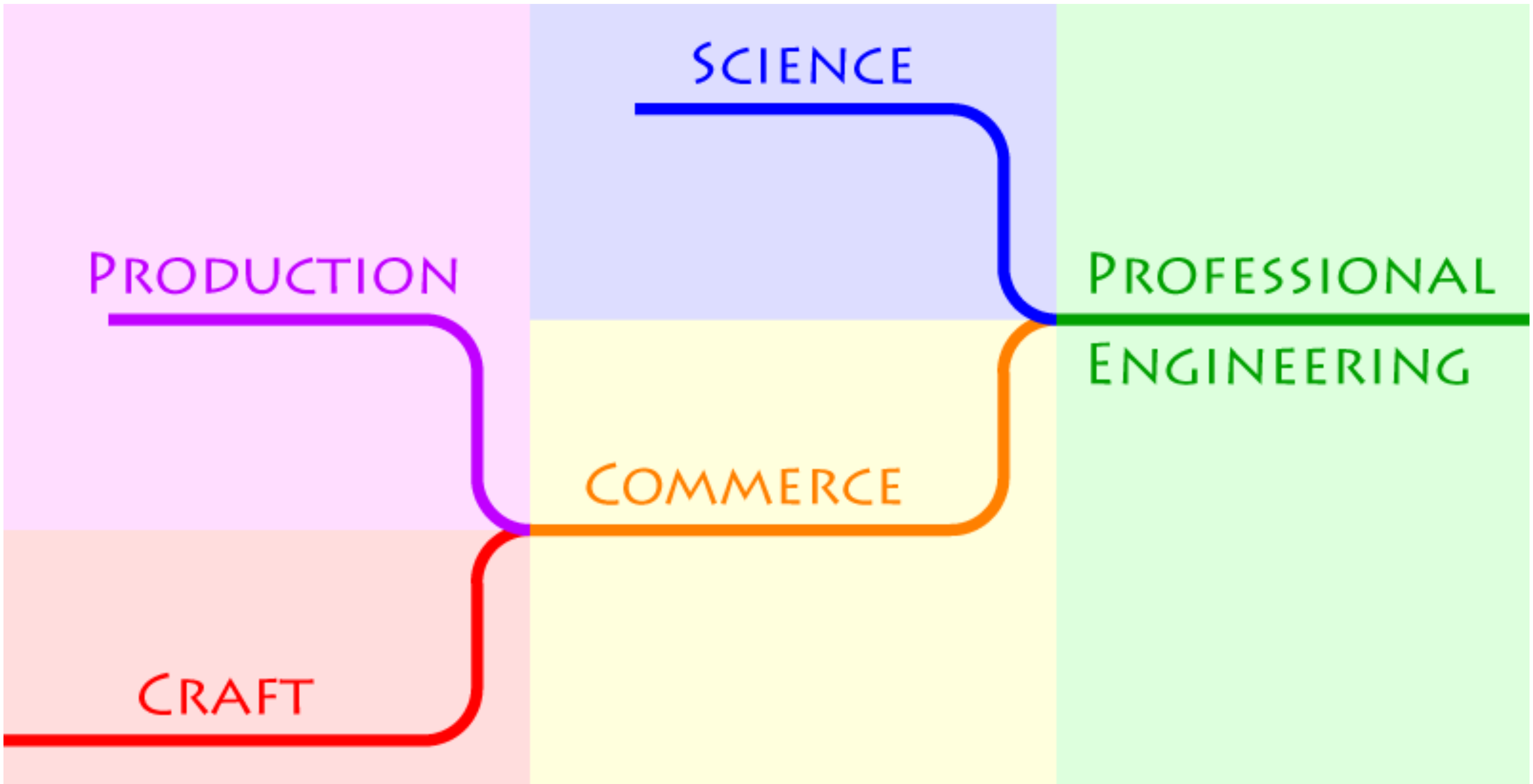
... building things ...

... 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

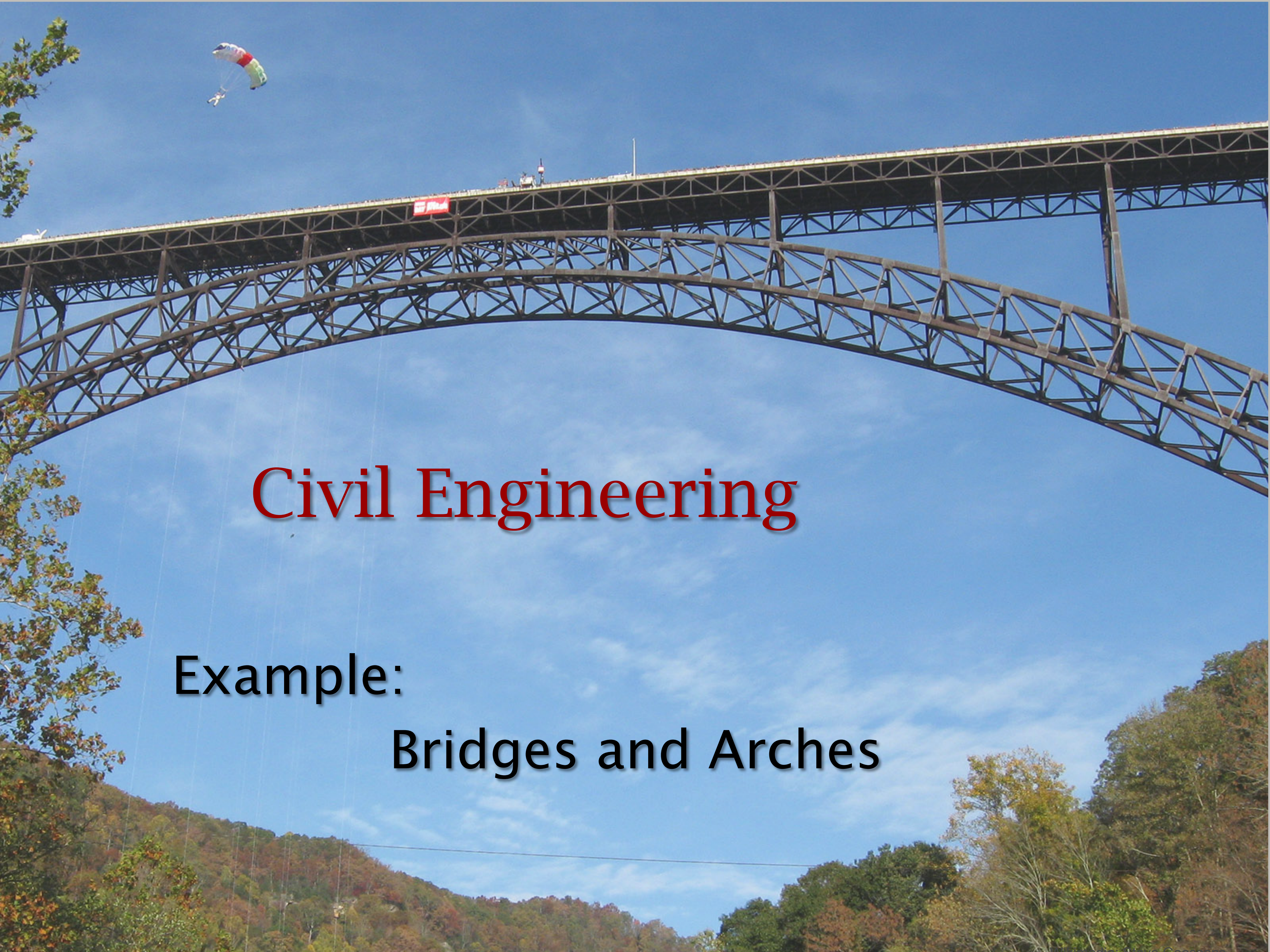

Science
Production
Professional
Engineering
Commerce
Craft

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 systematic, scientific knowledge.

Science often arises from progressive codification of practice.

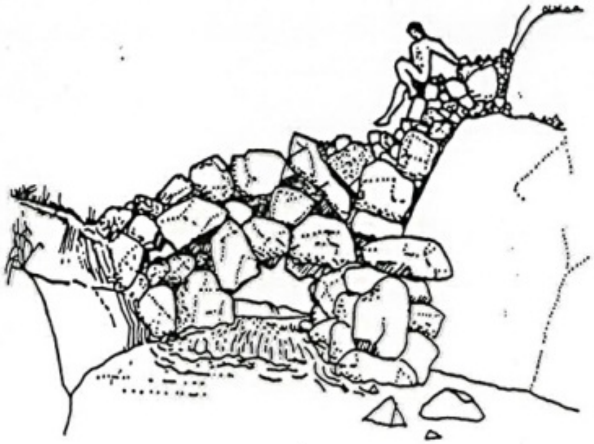
Civil Engineering as Model

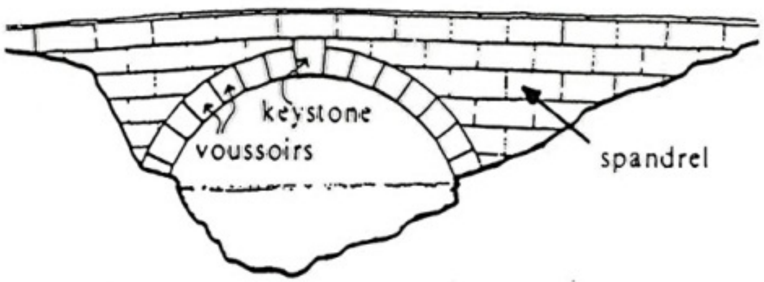


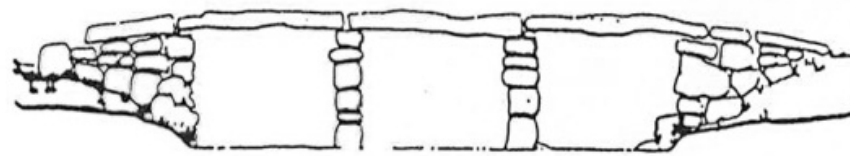
Civil Engineering

Example:

Bridges and Arches







<u>Great Buildings of the World</u>
<u>Bridges</u>, Derrick Beckett,
Hamlyn Publishing Group, Ltd.,
London, England, pp 10,12,16,19

1st Century CE

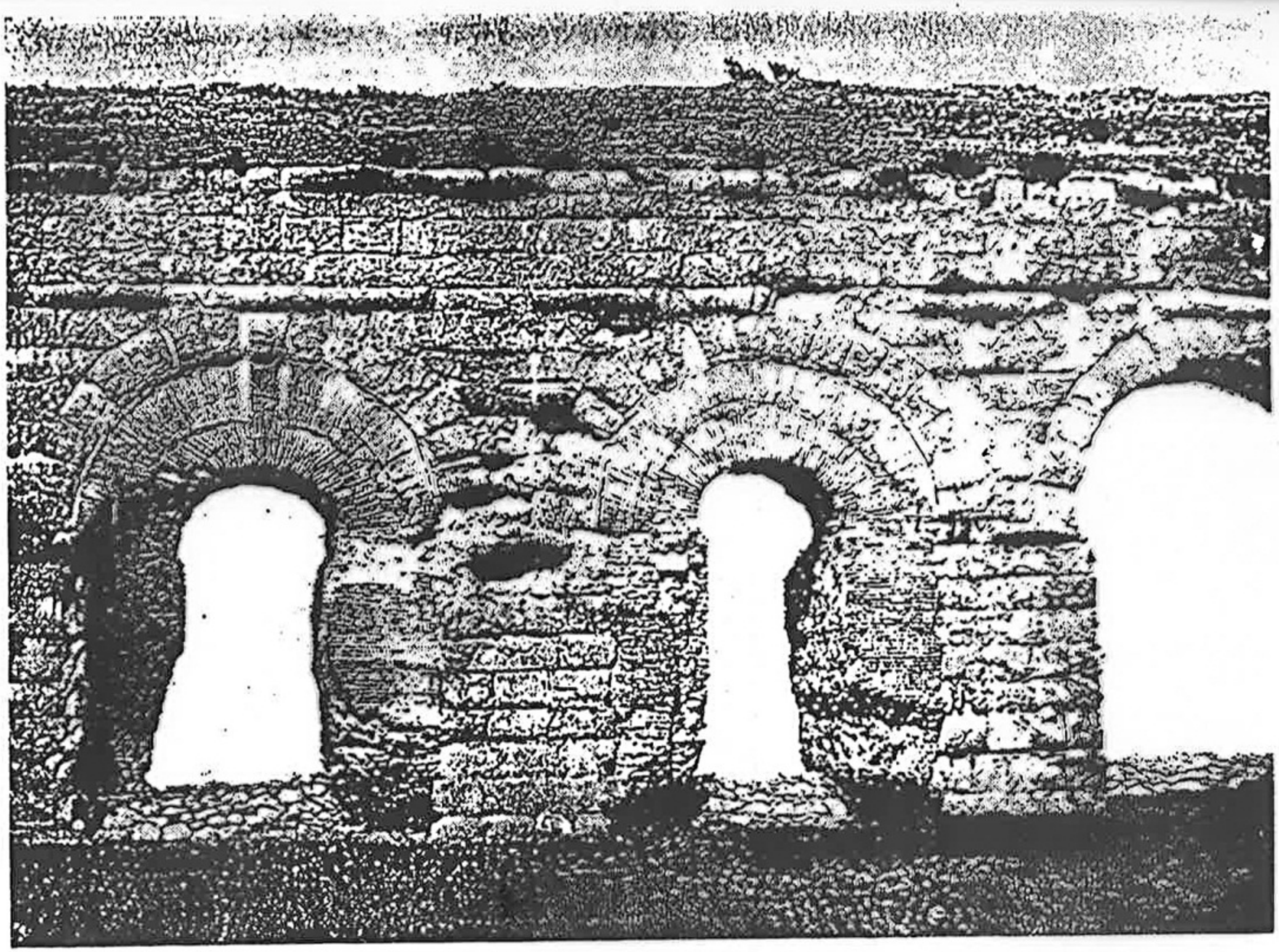


Figure 4.4 Two Roman aqueducts, Anio Novus built on Claudia (From Curt Merckel, *Die Ingenieurtechnik im Alterthum*, 1899; courtesy Julius Springer-Verlag)

Craft of bridges

Romans

Renaissance & Industrial Revolution

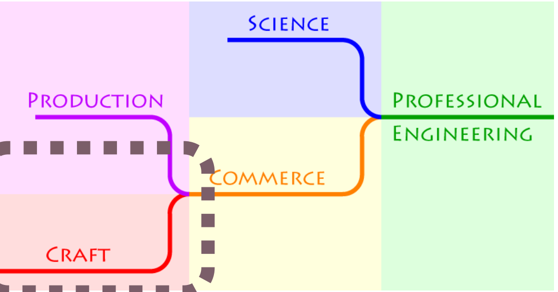
Scientific Engineering

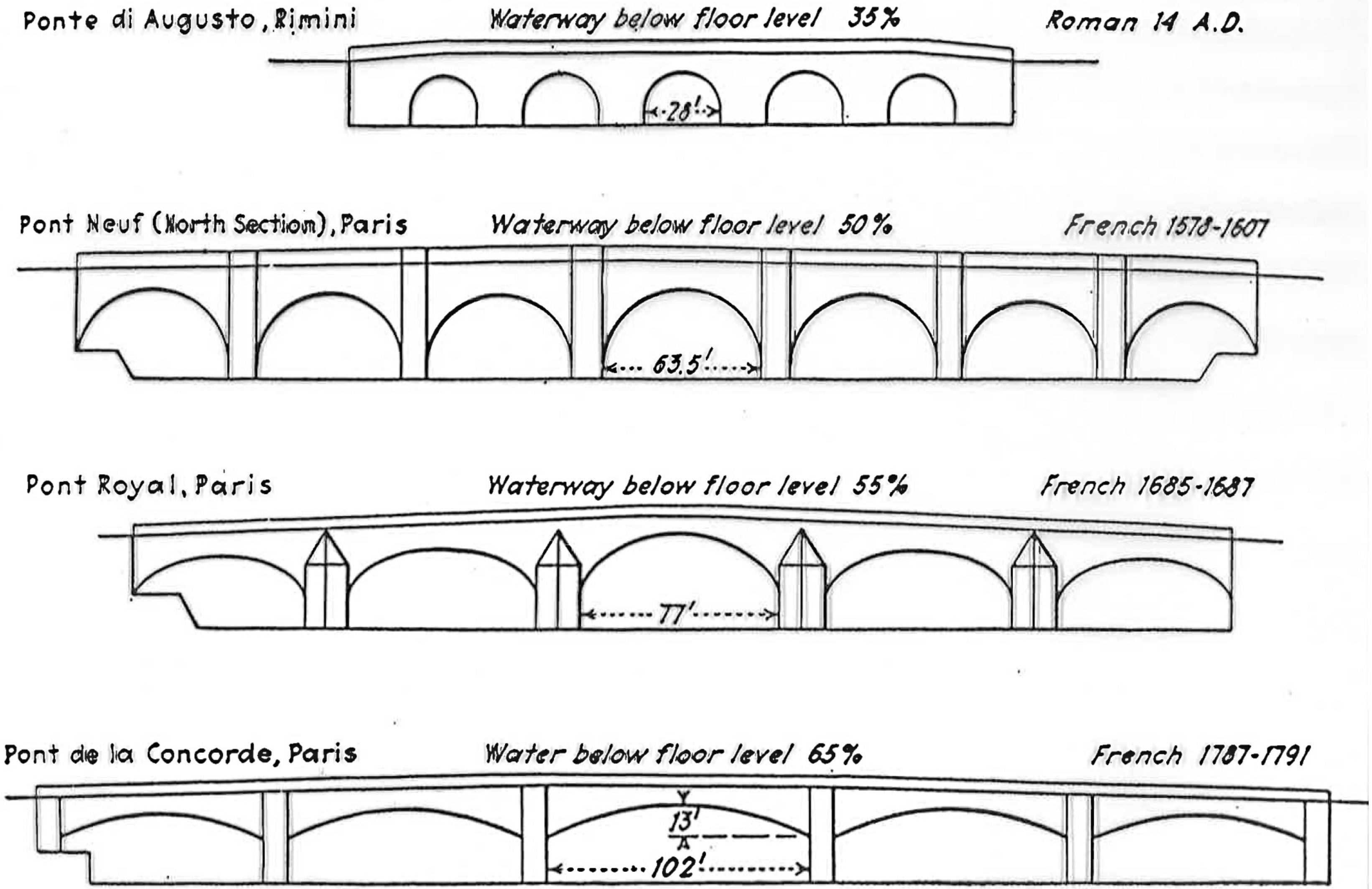
Empirical progress via failure and repair

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

ring and Western Civilization, James Kip Finch,
Hill Book Company, Inc., New York, NY, 1951, p33

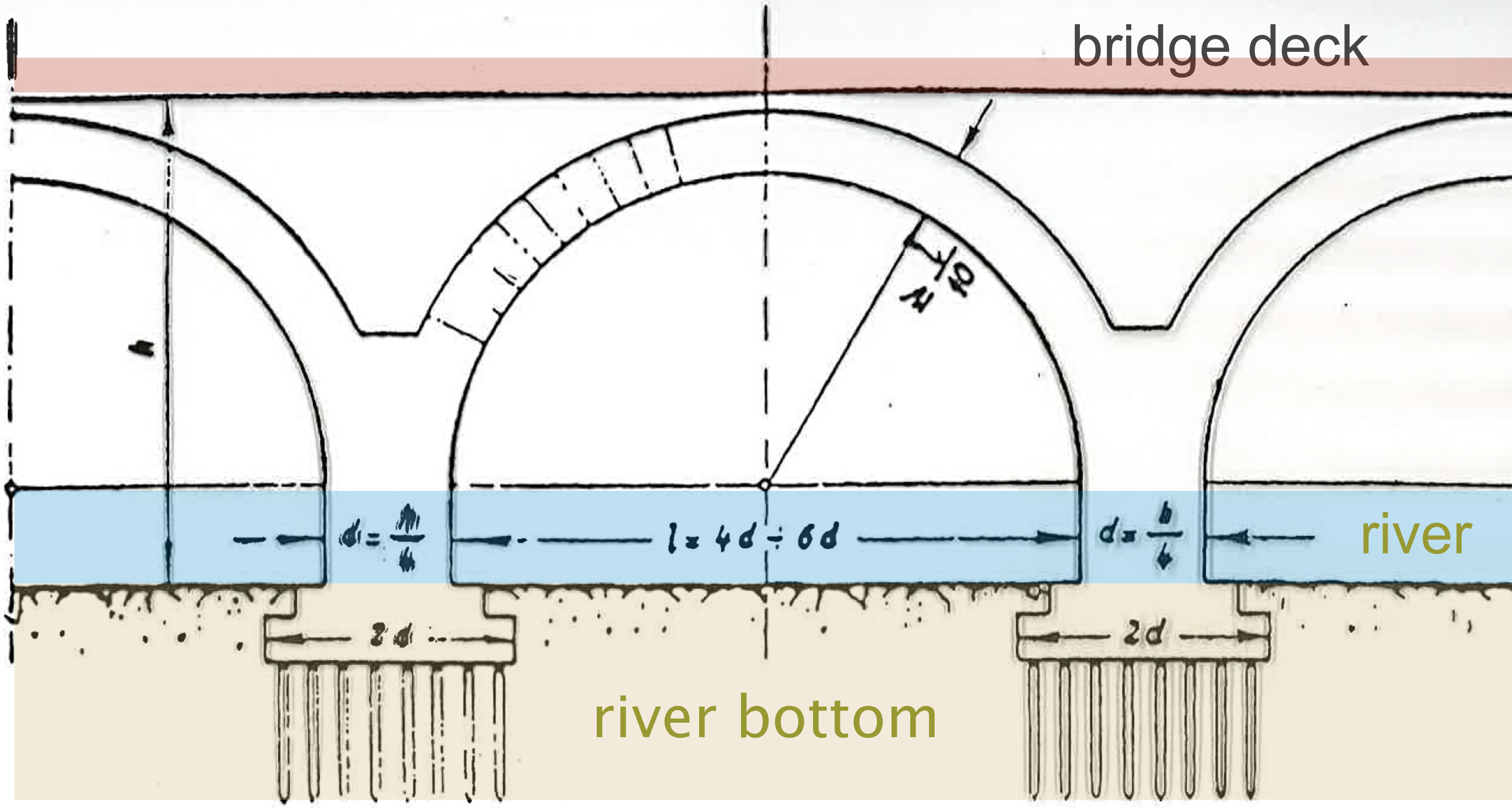
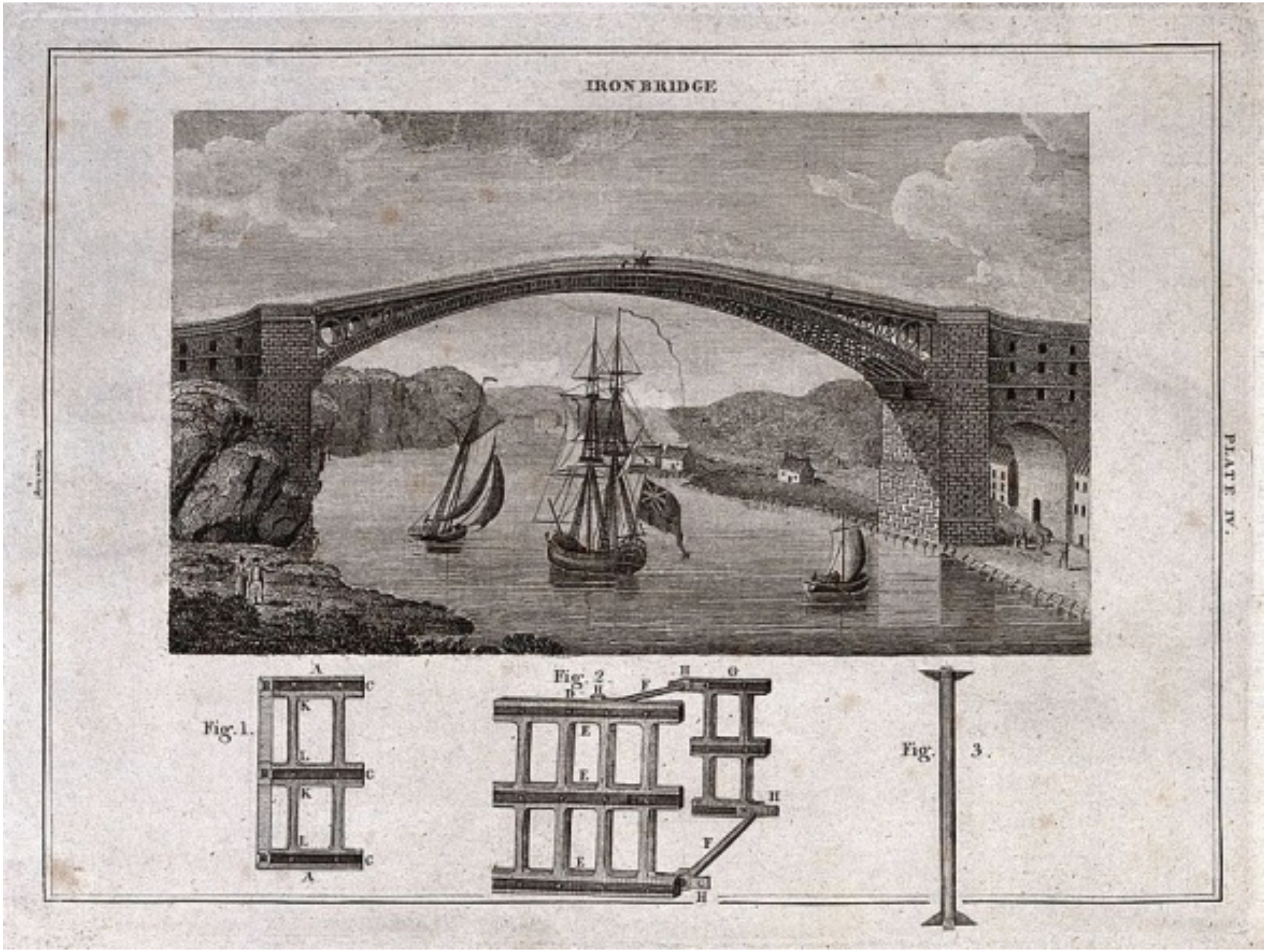


Fig. 28. Arch bridge, according to Leon Battista Alberti.

15th century

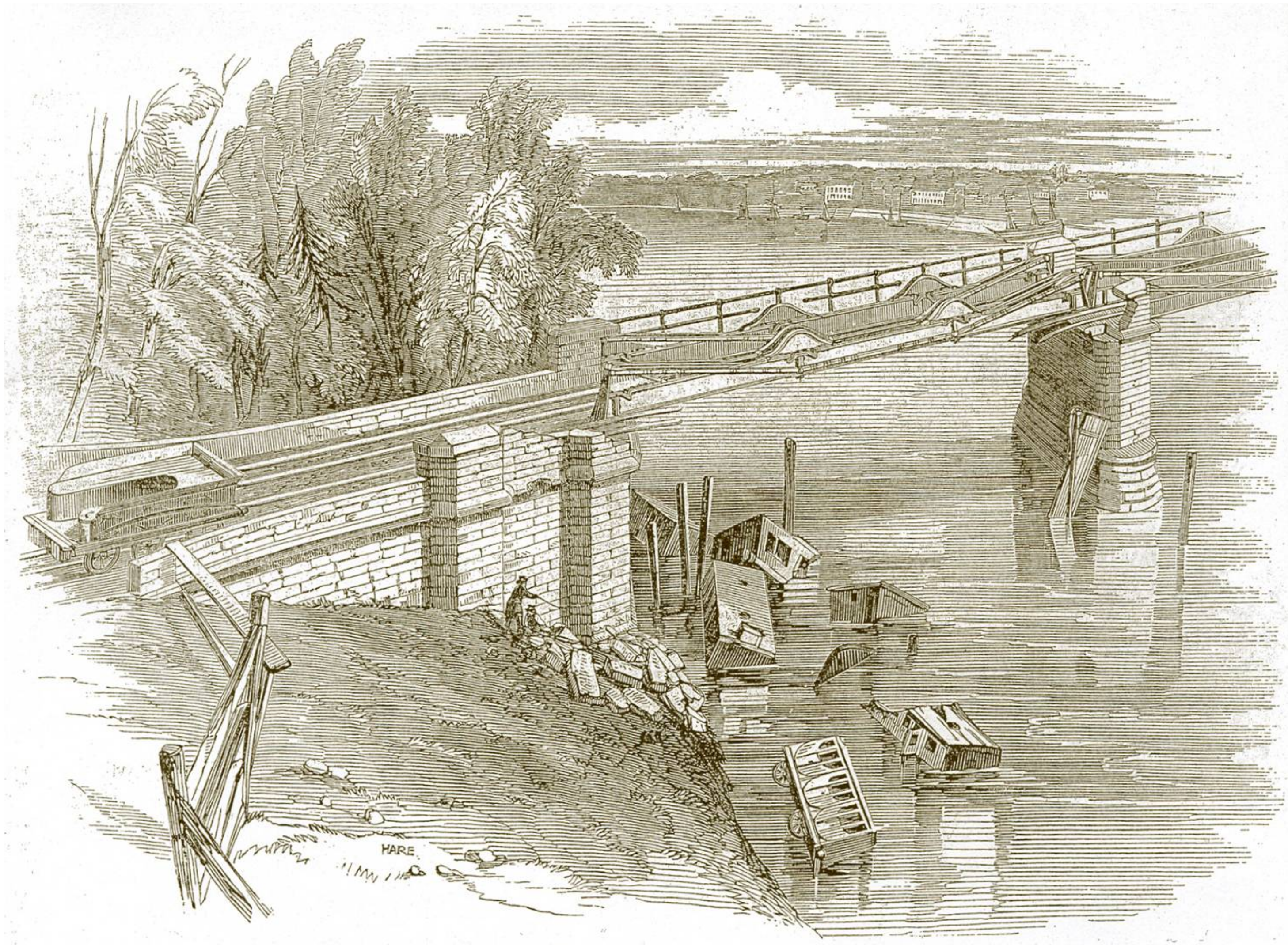
tory of Civil Engineering, Hans Straub,
.I.T. Press, Cambridge, MA, 1964, p90

Ironbridge at Coalbrookdale, 1779




Mary Shaw

Dee Bridge disaster, 1847



SCENE OF THE LATE RAILWAY ACCIDENT, AT CHESTER.—DILAPIDATED SPAN OF THE DEE BRIDGE.

Illustrated London News, 1847

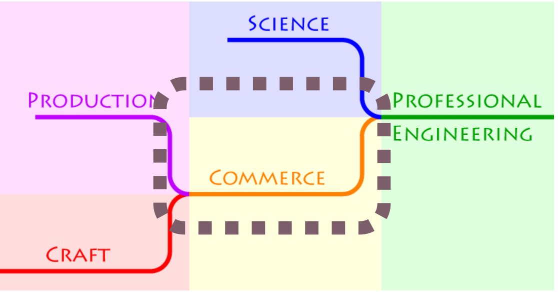
Business of bridges

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



Fundamental Problems

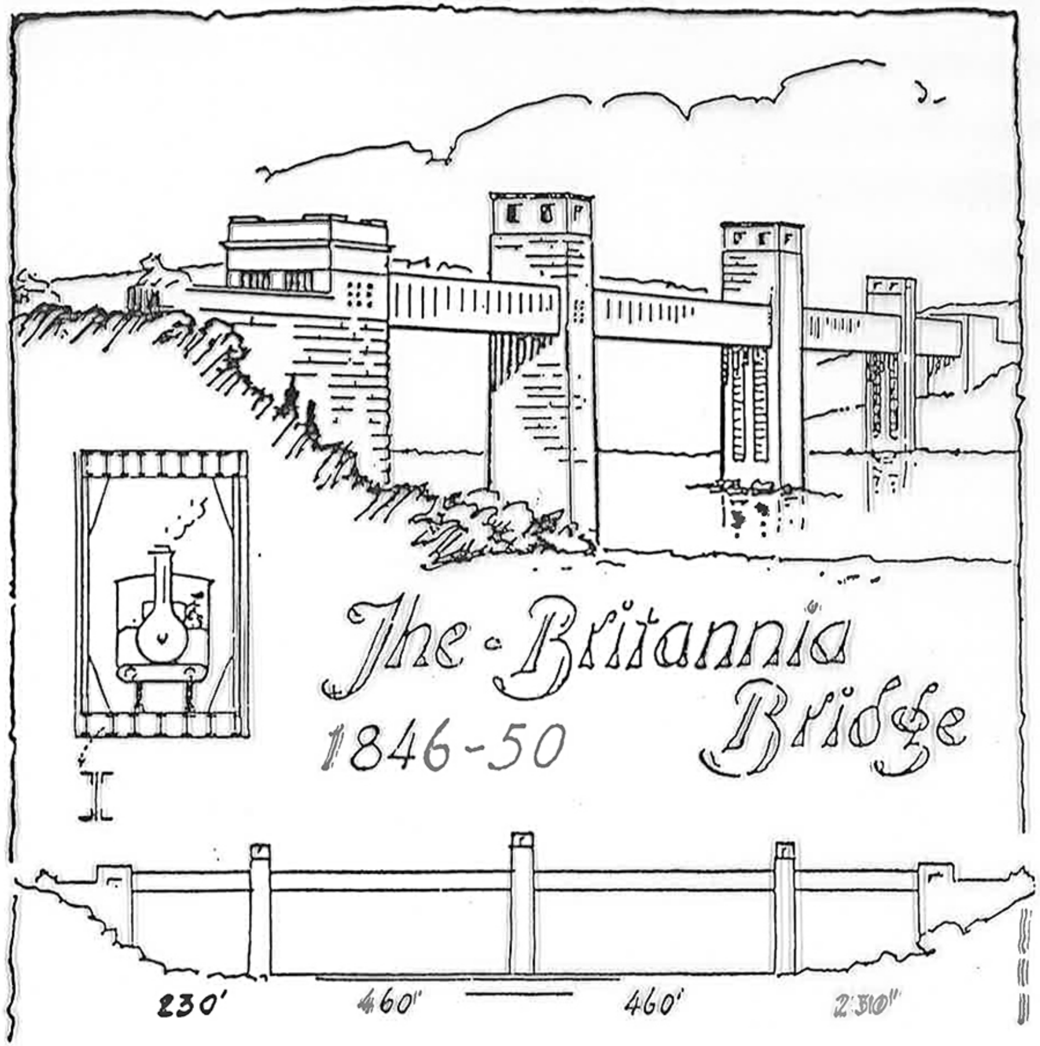
Composition of forces	Bending

Theories that solved these problems

Statics	Strength of materials
Varignon & Newton late 17th century	Coulomb & Navier early 18th century

Hardest problem was identifying the proper basic concepts, e.g. force.

New mathematics was needed (calculus).

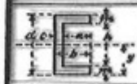
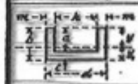
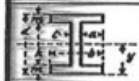
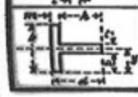


Story of Engineering, James Kip Finch
eday & Co., Inc., Garden City, NY,

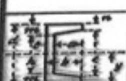
Wikimedia: Velela

...el Construction, American Institute of Steel Construction, Inc., January 1930, p.76 & 77

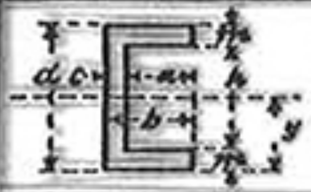
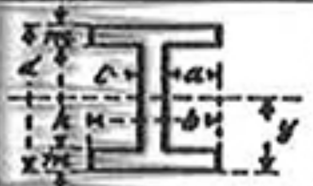
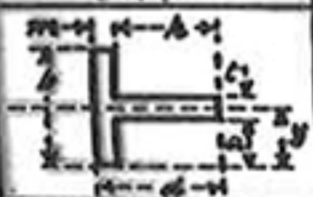
PROPERTIES OF VARIOUS SECTIONS

Sections	Area of Section A	Distance from Axis to Extremities of Section y and y_1	Moment of Inertia I	Section Modulus $S = \frac{I}{y}$	Radius of Gyration $r = \sqrt{\frac{I}{A}}$
	$bd - ah$	$y = \frac{d}{2}$	$\frac{1}{12}(bd^3 - ah^3)$	$\frac{bd^3 - ah^3}{6d}$	$\sqrt{\frac{bd^3 - ah^3}{12(bd - ah)}}$
	$bd - ah$	$y = b - y_1$ $y_1 = \frac{2b^2m + ht^2}{2A}$	$\frac{1}{3}(2mb^3 + ht^3) - Ay_1^2$	$\frac{I}{y}$	$\sqrt{\frac{I}{A}}$
	$bd - 2ah$	$y = \frac{d}{2}$	$\frac{1}{12}(bd^3 - 2ah^3)$	$\frac{bd^3 - 2ah^3}{6d}$	$\sqrt{\frac{I}{A}}$
	$bd - 2ah$	$y = \frac{b}{2}$	$\frac{1}{12}(2mb^3 + ht^3)$	$\frac{2mb^3 + ht^3}{6b}$	$\sqrt{\frac{I}{A}}$
	$bm + ht$	$y = d - y_1$ $y_1 = \frac{d^2t + m^2(b - t)}{2A}$	$\frac{1}{3}\left(ty^3 + by_1^3 - a(y_1 - m)^3\right)$	$\frac{I}{y}$	$\sqrt{\frac{I}{A}}$
	$bm + ht$	$y = \frac{b}{2}$	$\frac{1}{12}(mb^3 + ht^3)$	$\frac{mb^3 + ht^3}{6b}$	$\sqrt{\frac{I}{A}}$

PROPERTIES OF VARIOUS SECTIONS

Sections	Area of Section A	Distance from Axis to Extremities of Section y and y_1	Moment of Inertia I	Section Modulus $S = \frac{I}{y}$	Radius of Gyration $r = \sqrt{\frac{I}{A}}$
	$bd + a(m + n)$	$y = \frac{d}{2}$	$\frac{1}{12}\left[bd^3 - \frac{a}{8(m - n)}(e^4 - h^4)\right]$	$\frac{2I}{d}$	$\sqrt{\frac{I}{A}}$
	$bd + a(m + n)$	$y = b - y_1$ $y_1 = \frac{b^2n + \frac{at^2}{3} + \frac{a(m - n)}{3}(b + 2t)}{A}$	$\frac{1}{3}\left[2nb^3 + ht^3 + \frac{m - n}{2a}(b^4 - t^4)\right] - Ay_1^2$	$\frac{I}{y}$	$\sqrt{\frac{I}{A}}$
	$bd + 2a(m + n)$	$y = \frac{d}{2}$	$\frac{1}{12}\left[bd^3 - \frac{a}{4(m - n)}(e^4 - h^4)\right]$	$\frac{2I}{d}$	$\sqrt{\frac{I}{A}}$
	$bd + 2a(m + n)$	$y = \frac{b}{2}$	$\frac{1}{12}\left[2nb^3 + ht^3 + \frac{m - n}{4a}(b^4 - t^4)\right]$	$\frac{2I}{b}$	$\sqrt{\frac{I}{A}}$
	$\frac{a(t + u)}{2} + tm + a(m + n)$	$y = h - y_1$ $y_1 = [6am^2 + 2a(m - n)(m + 2n) + 3td^2 - a(t - u)(3d - a)] \div 6A$	$\frac{1}{12}\left[a^3(3u + t) + 4bm^3 - 2a(m - n)^3\right] - A(y_1 - m)^2$	$\frac{I}{y}$	$\sqrt{\frac{I}{A}}$
	$\frac{a(t + u)}{2} + tm + a(m + n)$	$y = \frac{b}{2}$	$\frac{nb^3 + (m - n)t^3 + au^3}{12}$ $+ \frac{a(m - n)[2a^2 + (2a + 3t)^2]}{36}$ $+ \frac{a(t - u)[(t - u)^2 + 2(t + 2u)^2]}{144}$	$\frac{2I}{b}$	$\sqrt{\frac{I}{A}}$

PROPERTIES OF VARIOUS SECTIONS

Sections	Area of Section A	Distance from Axis to Extremities of Section y and y_1	Moment of Inertia I	Section Modulus $S = \frac{I}{y}$
	$bd - ah$	$y = \frac{d}{2}$	$\frac{1}{12}(bd^3 - ah^3)$	$\frac{bd^3 - ah^3}{6d}$
	$bd - ah$	$y = b - y_1$ $y_1 = \frac{2b^2m + ht^2}{2A}$	$\frac{1}{3}(2mb^3 + ht^3) - Ay_1^2$	$\frac{I}{y}$
	$bd - 2ah$	$y = \frac{d}{2}$	$\frac{1}{12}(bd^3 - 2ah^3)$	$\frac{bd^3 - 2ah^3}{6d}$
	$bd - 2ah$	$y = \frac{b}{2}$	$\frac{1}{12}(2mb^3 + ht^3)$	$\frac{2mb^3 + ht^3}{6b}$
	$bm + ht$	$y = d - y_1$ $y_1 = \frac{d^2t + m^2(b - t)}{2A}$	$\frac{1}{3}\left(ty^3 + by_1^3 - 2a(y_1 - m)^3\right)$	$\frac{I}{y}$
	$bm + ht$	$y = \frac{b}{2}$	$\frac{1}{12}(mb^3 + ht^3)$	$\frac{mb^3 + ht^3}{6b}$

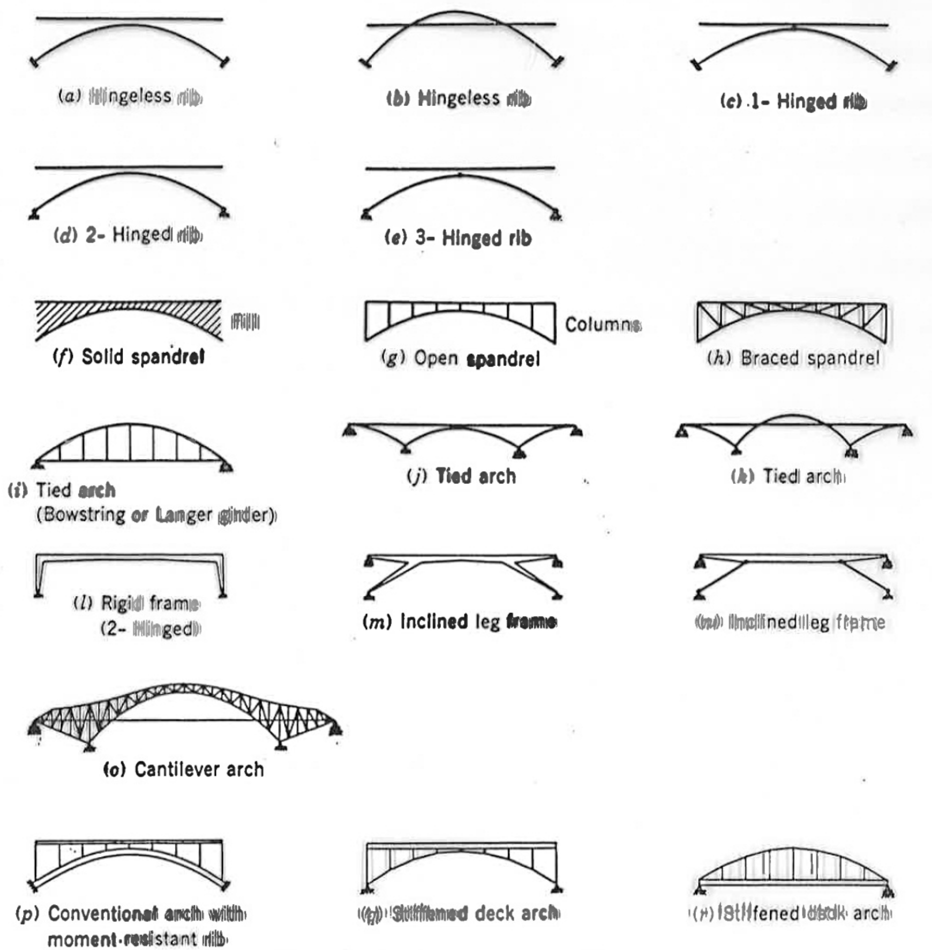


Figure 10.2 Types of arch bridge.

ign of Bridge Superstructures, Colin O'Connor,
y-Interscience, New York, NY, 1971, p. 489

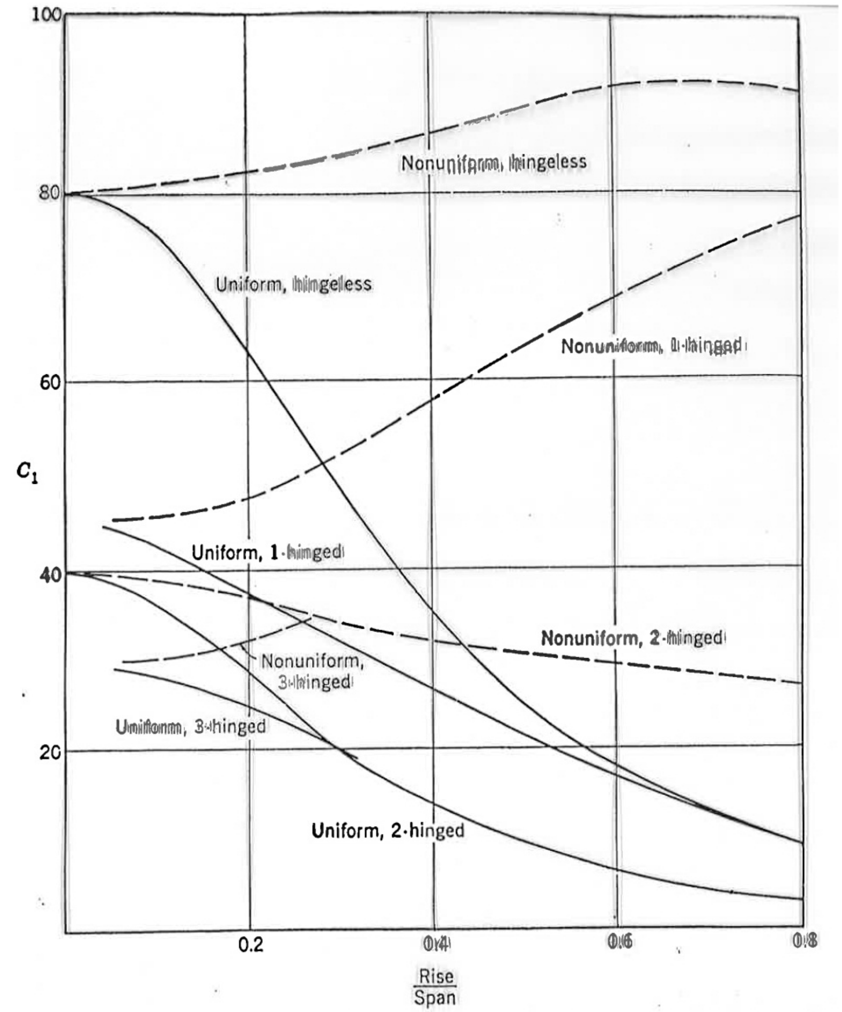


Figure 10.29 Coefficients for in-plane buckling of parabolic arch [59] $H_{cr} = C_1(EI/L^2)$.

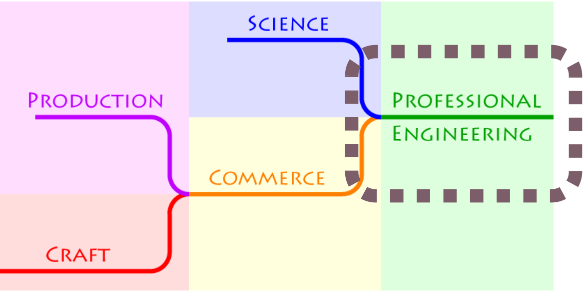
ign of Bridge Superstructures, Colin O'Connor,
y-Interscience, New York, NY, 1971,

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.
- http://bradd.engrprograms.com/home/

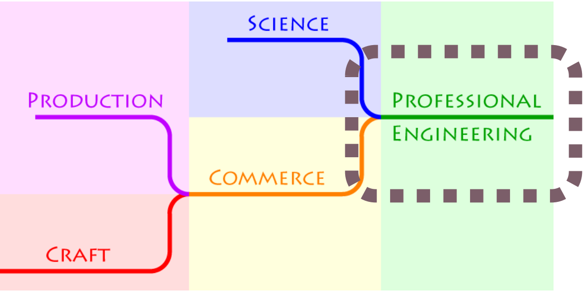
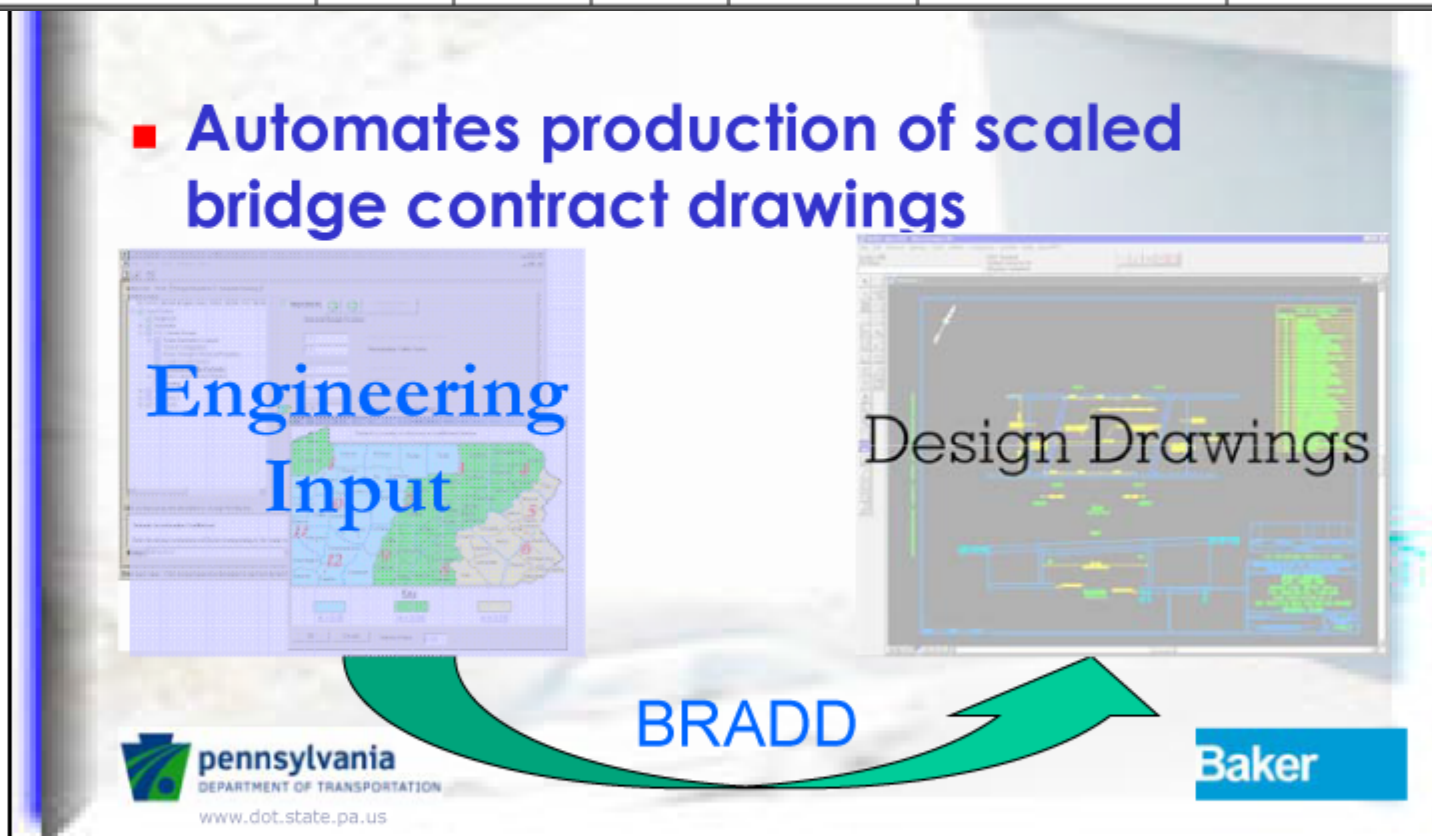
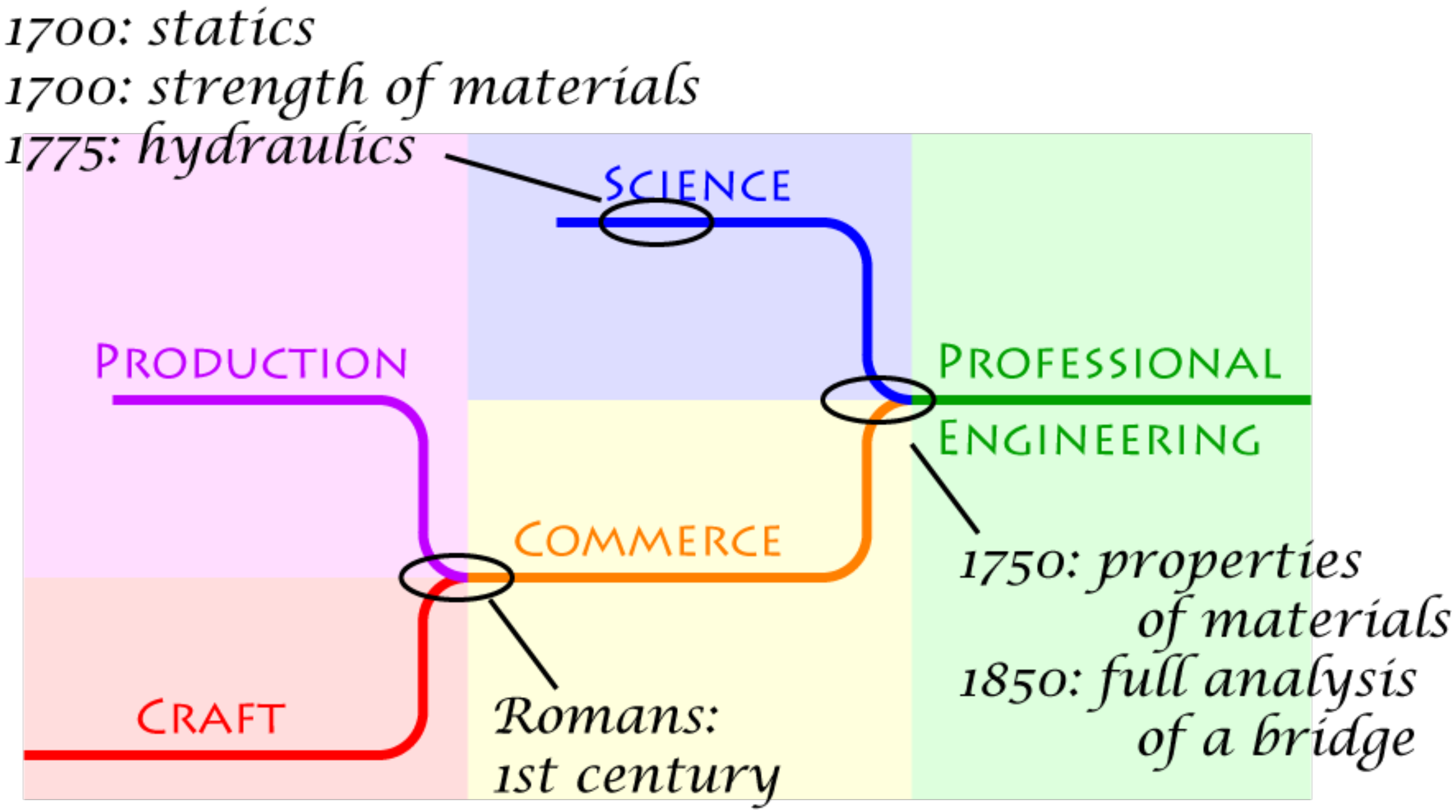


Table 2.3-2 Matrix of Abutment Types versus Superstructure Types

Superstructure Type	Abutment Type					
	Traditional			Integral	SuperOnly High/Stub/Wall	SuperOnly Integral
	High	Wall	Stub			
Prestressed Concrete Adjacent Box Beam		☒			☒	
Prestressed Concrete Spread Box Beam	☒	☒	☒	☒	☒	☒
Prestressed Concrete I-Beam	☒	☒	☒	☒	☒	☒
Steel Rolled Beam	☒	☒	☒	☒	☒	☒
Steel Plate Girder	☒	☒	☒	☒	☒	☒



Evolution of civil engineering



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

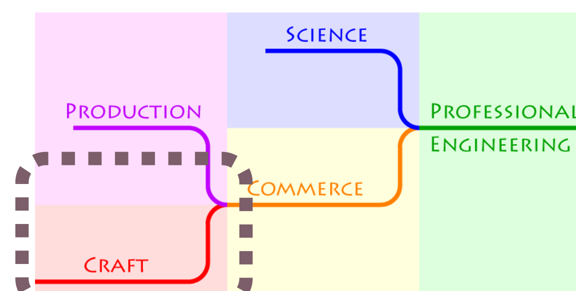
"Software Engineering"

Rallying Cry

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

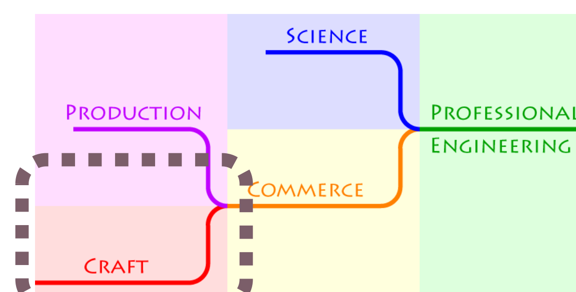
Aspiration, not description

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

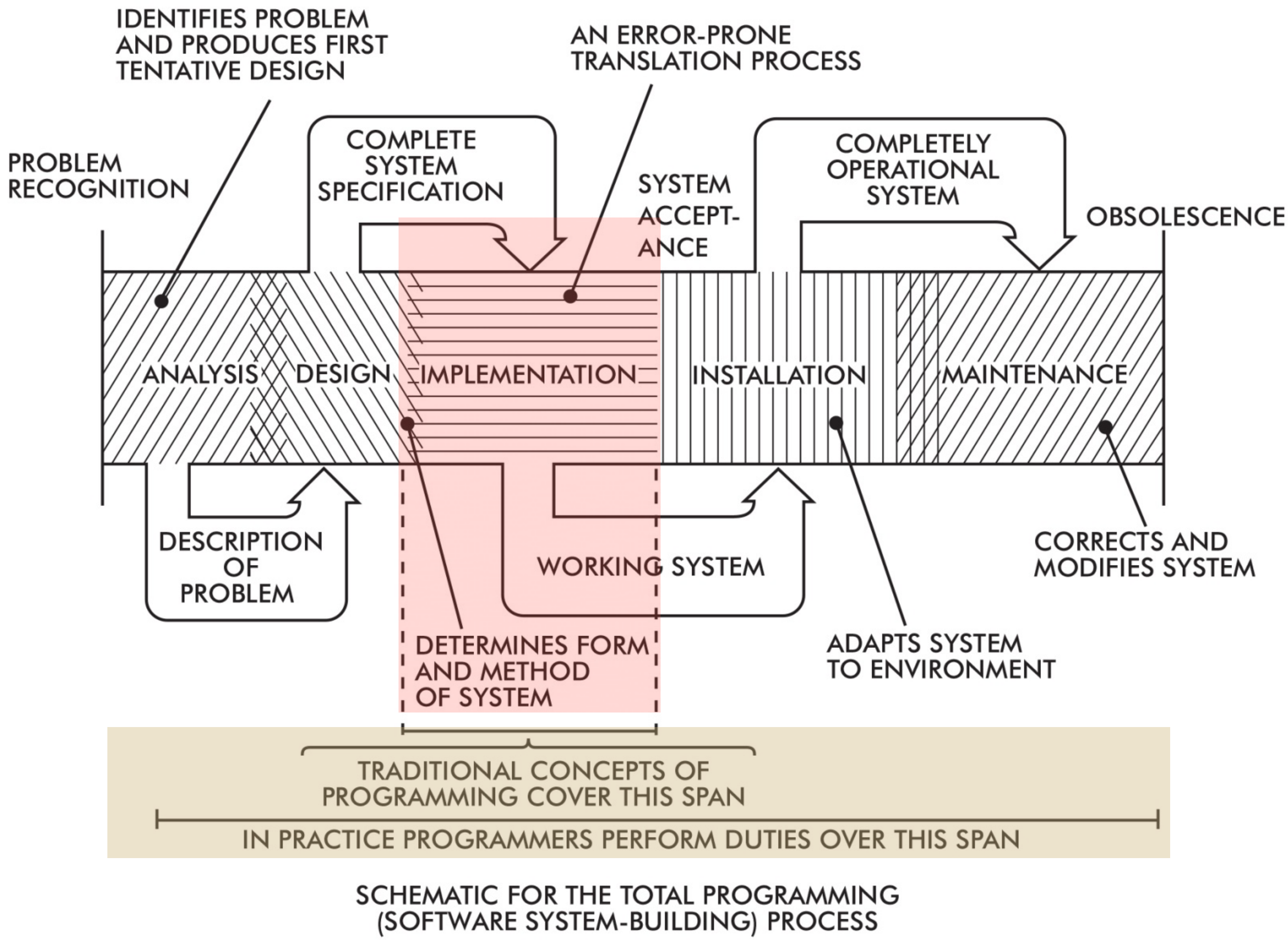
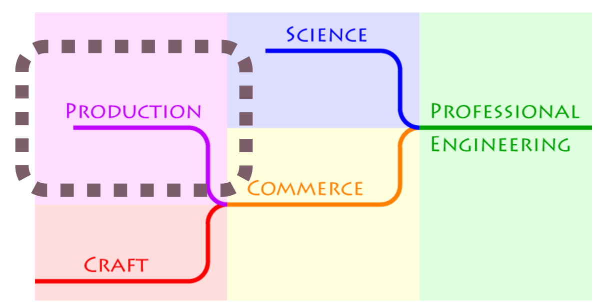


Figure 2. From Selig: Documentation for service and users. Originally due to Constantine.

Production techniques

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

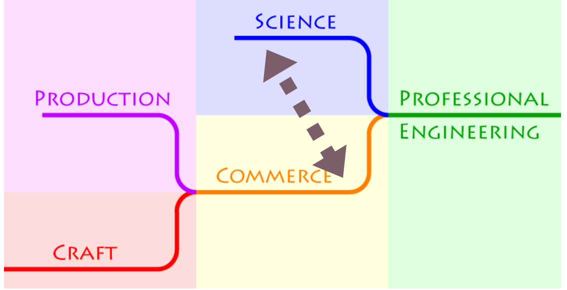
- Structured programming
- Waterfall models
- Incremental and iterative development
- Cost/schedule estimation
- Process maturity
- Extreme, agile processes



Commerce 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



Codified knowledge

Data structures, algorithms

Programming languages and semantics

Verification and model checking

Objects and abstract data types

Static and dynamic analysis

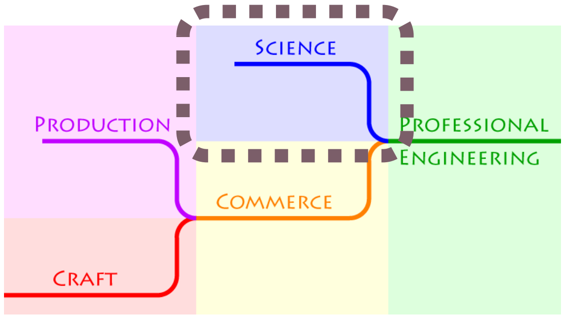
Software architectures

Model-based engineering

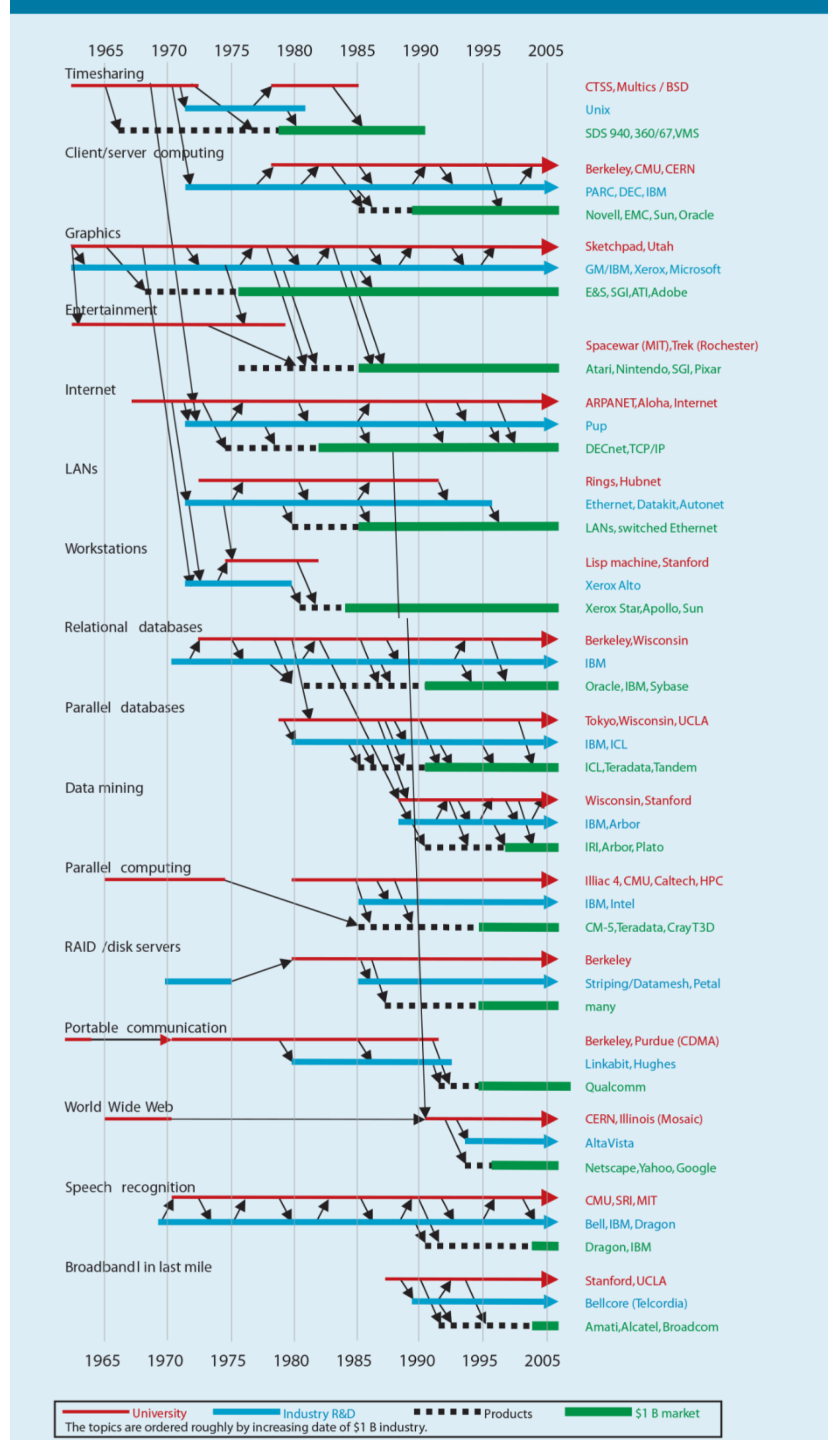
Pattern languages

Computability

. . .

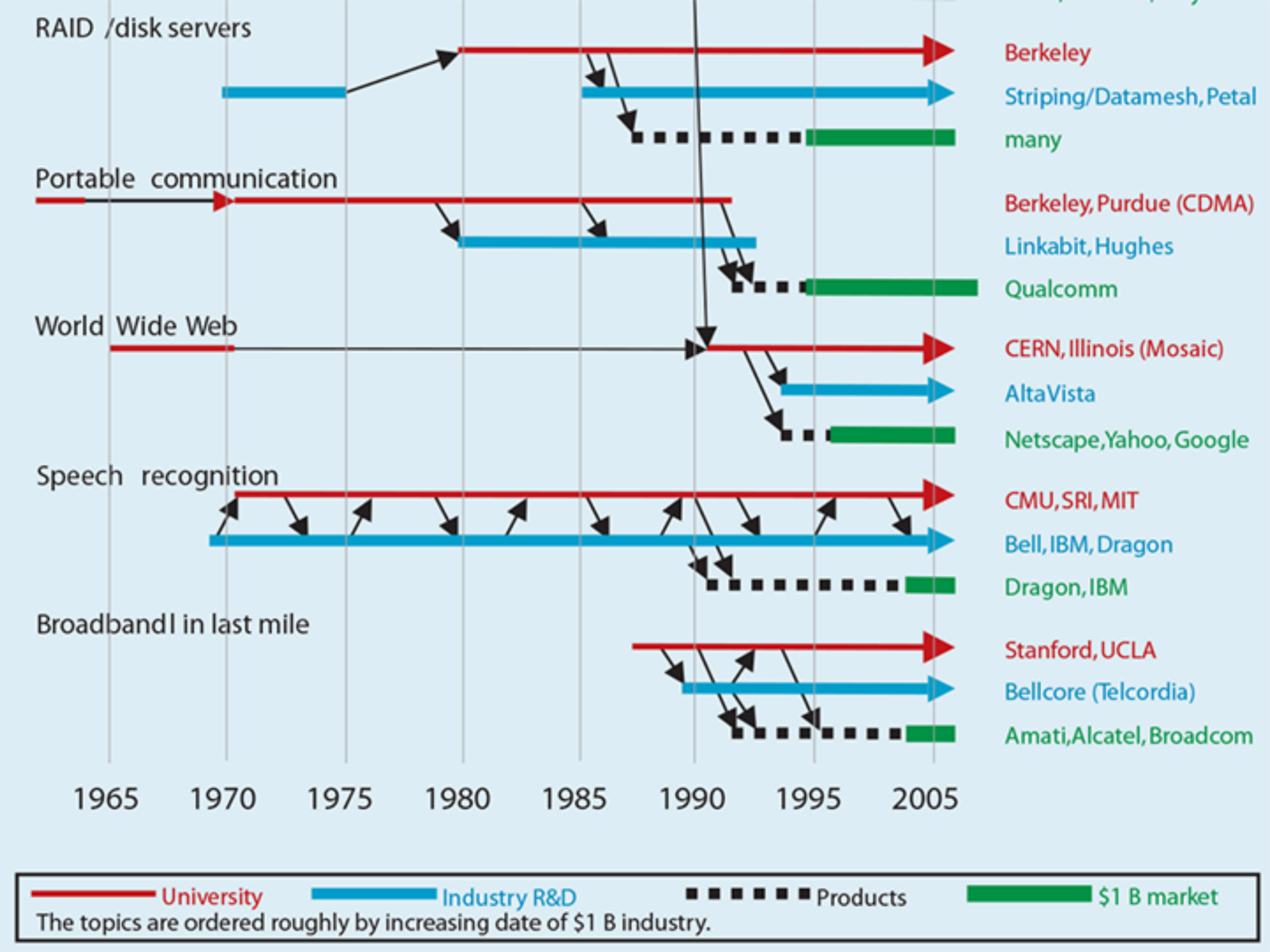


Research feeds practice

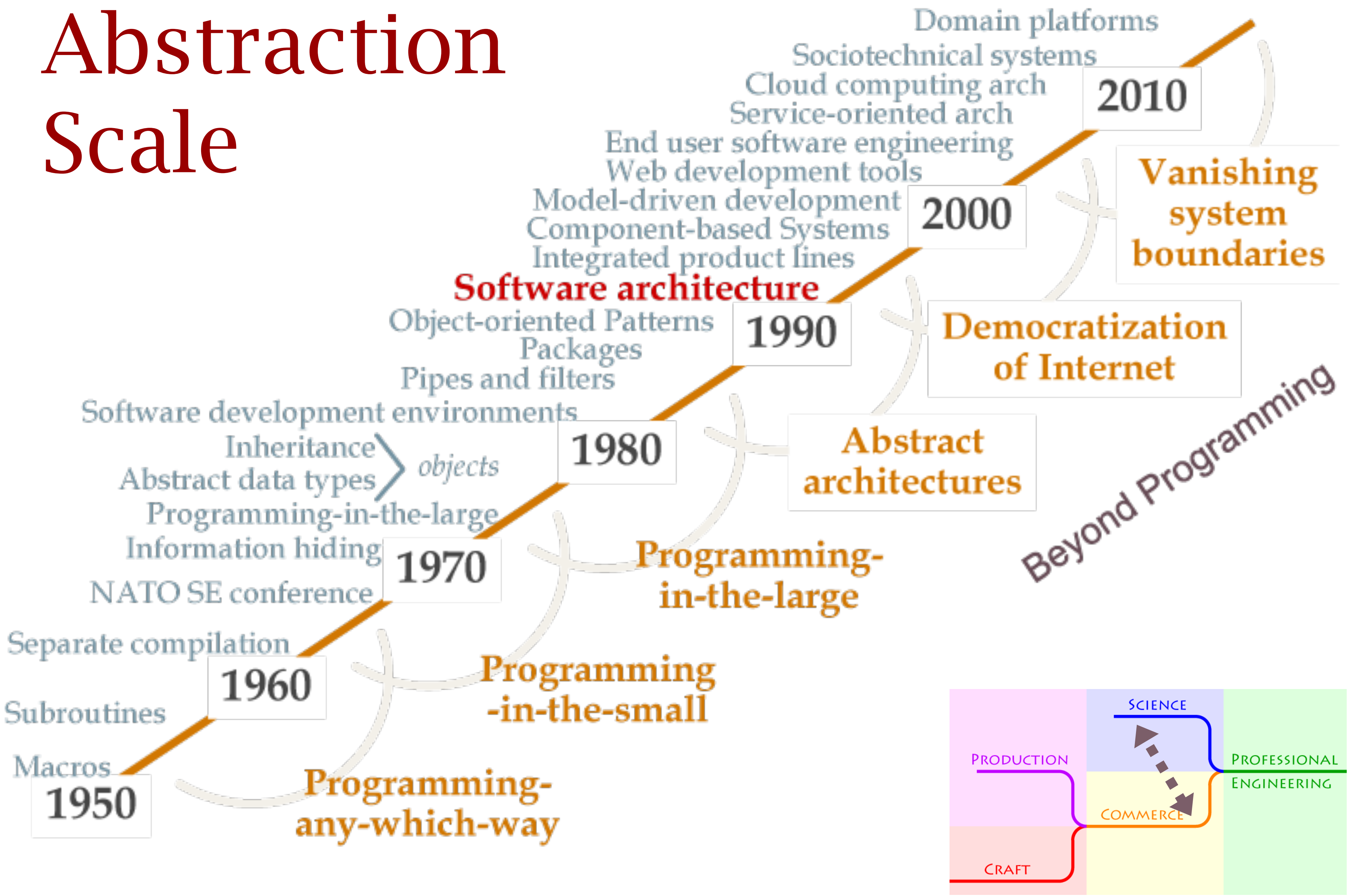


Research and development stimulates creation of innovative ideas and industries.

E. Lazowska.
Viewpoint.
CACM Aug 2008


RAID /disk servers
Berkeley
Striping/Datamesh, Petal
many
Portable communication
Berkeley, Purdue (CDMA)
Linkabit, Hughes
Qualcomm
World Wide Web
CERN, Illinois (Mosaic)
AltaVista
Netscape, Yahoo, Google
Speech recognition
CMU, SRI, MIT
Bell, IBM, Dragon
Dragon, IBM
Broadbandl in last mile
Stanford, UCLA
Bellcore (Telcordia)
Amati, Alcatel, Broadcom
1965
1970
1975
1980
1985
1990
1995
2005
University
Industry R&D
Products
$1 B market
The topics are ordered roughly by increasing date of $1 B industry.

Increasing Abstraction Scale



Design guidance

Choosing among algorithms based on the problem setting

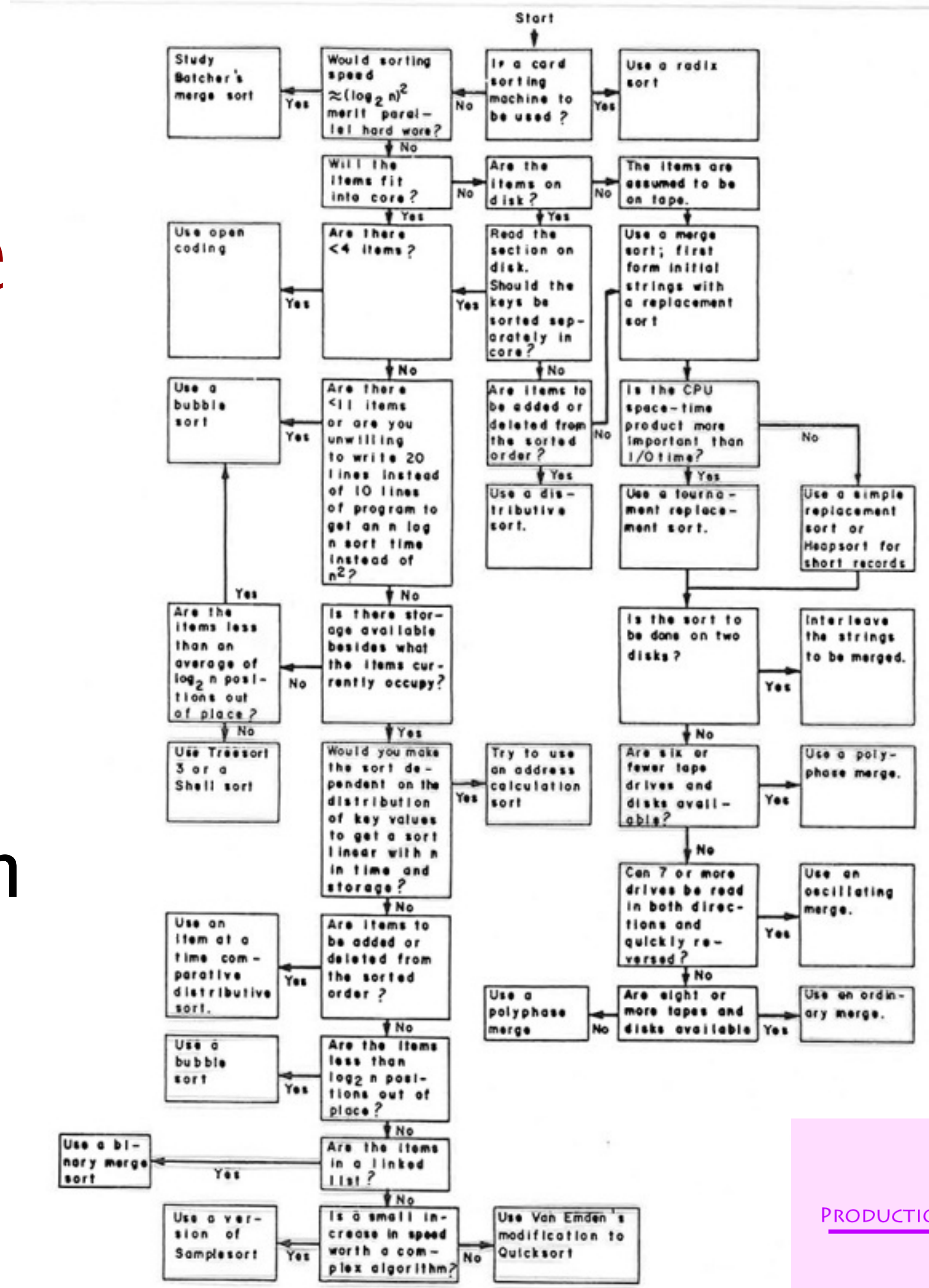
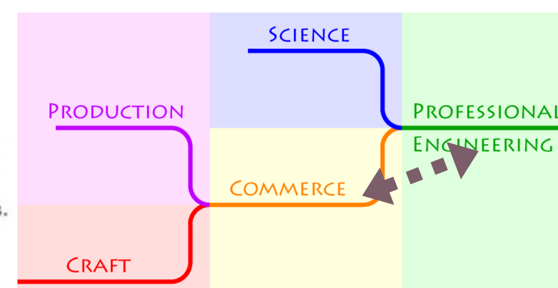
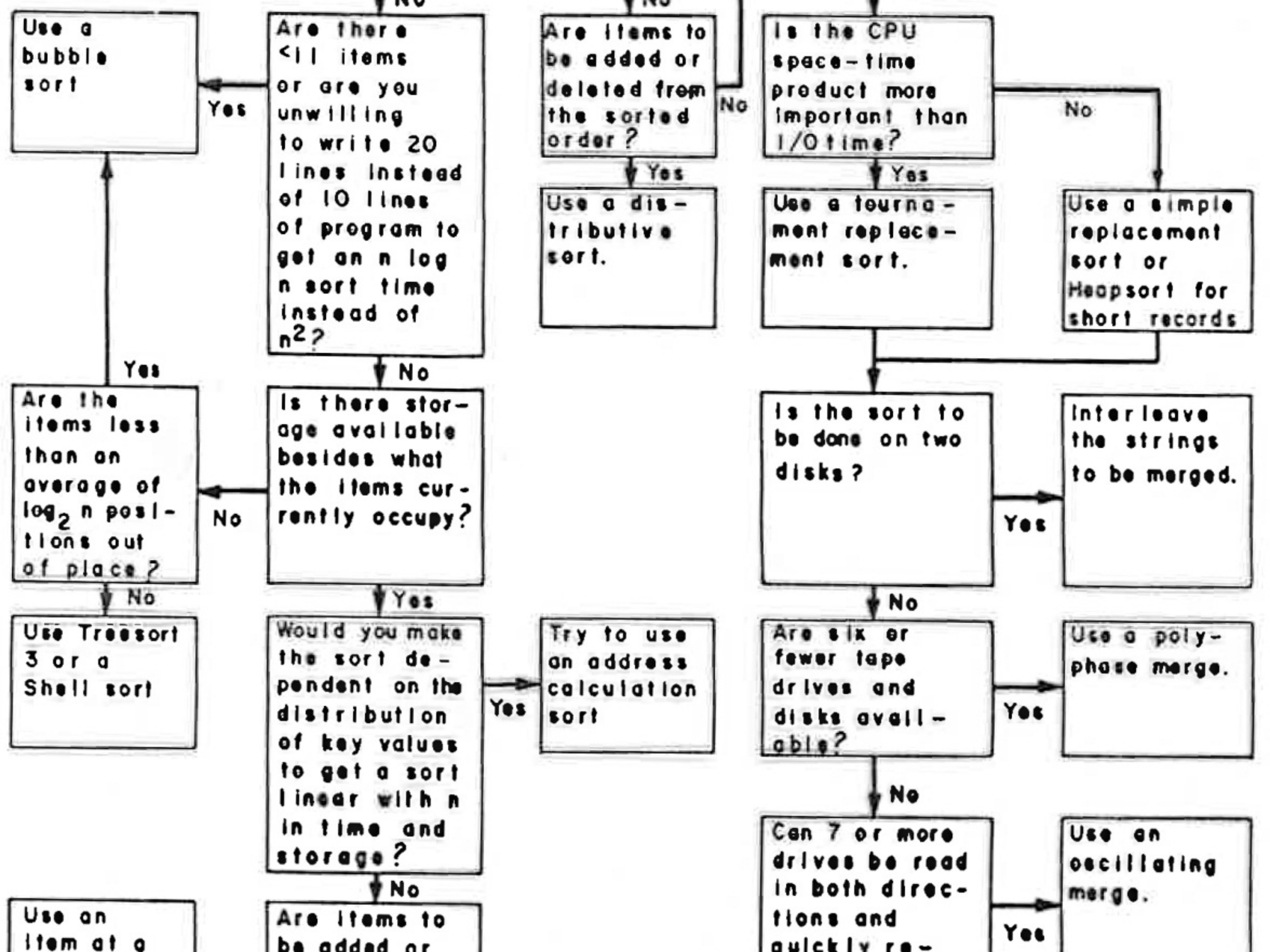


Fig. 21. Suggestions on the use of the various sorting techniques.




No
No
Use a bubble sort
Yes
Are there <11 items or are you unwilling to write 20 lines instead of 10 lines of program to get an n log n sort time instead of n²?
No
Are items to be added or deleted from the sorted order?
No
Yes
Use a distributive sort.
Is the CPU space-time product more important than I/O time?
No
Yes
Use a tournament replacement sort.
Use a simple replacement sort or Heapsort for short records
Yes
No
Are the items less than an average of log₂ n positions out of place?
No
Is there storage available besides what the items currently occupy?
Is the sort to be done on two disks?
Yes
Interleave the strings to be merged.
No
Yes
Use Treesort 3 or a Shell sort
Would you make the sort dependent on the distribution of key values to get a sort linear with n in time and storage?
Yes
Try to use an address calculation sort
No
Are six or fewer tape drives and disks available?
Yes
Use a polyphase merge.
No
No
Use an item at a
Are items to be added or
Can 7 or more drives be read in both directions and quickly re-
Yes
Use an oscillating merge.

Software Architecture

Software architecture …

- … is principled understanding of the large-scale structure of software systems as collections of interacting elements
- … 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

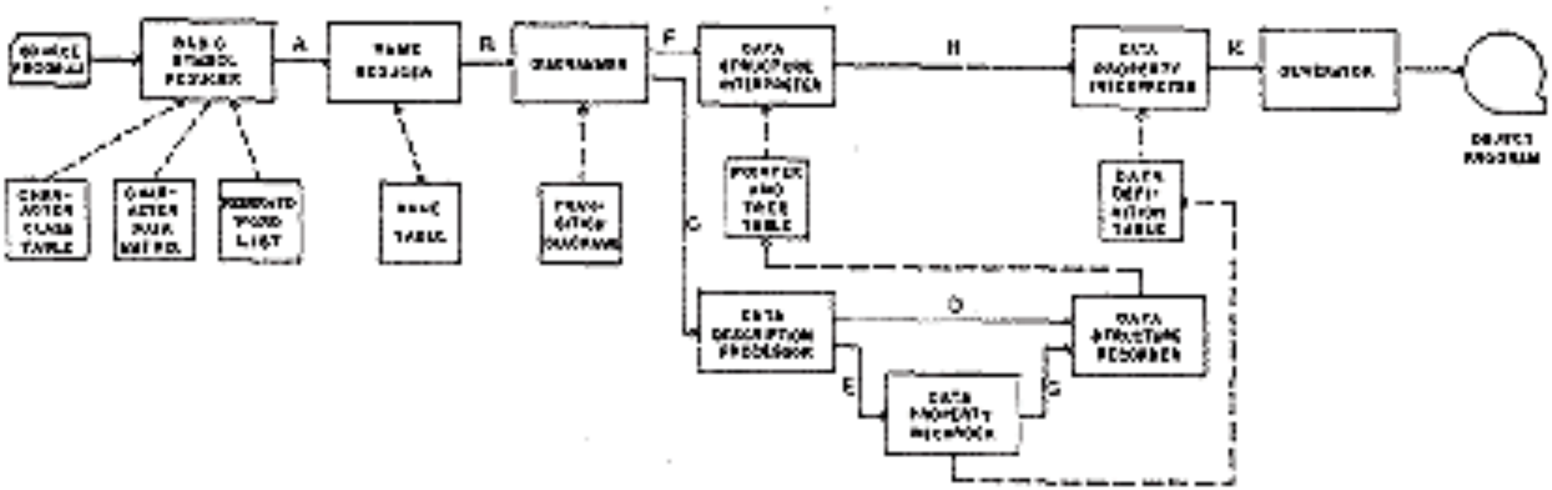


Fig. 4. COBOL Compiler Organization

with a program transformation

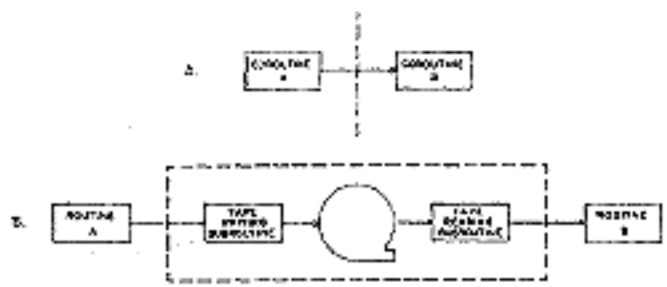
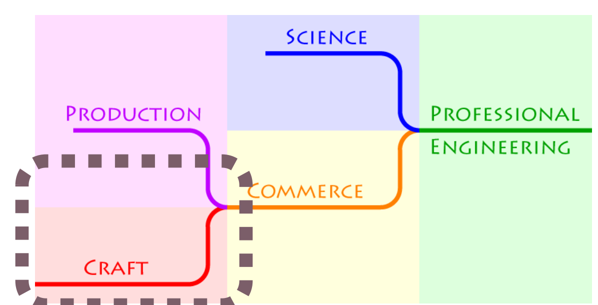
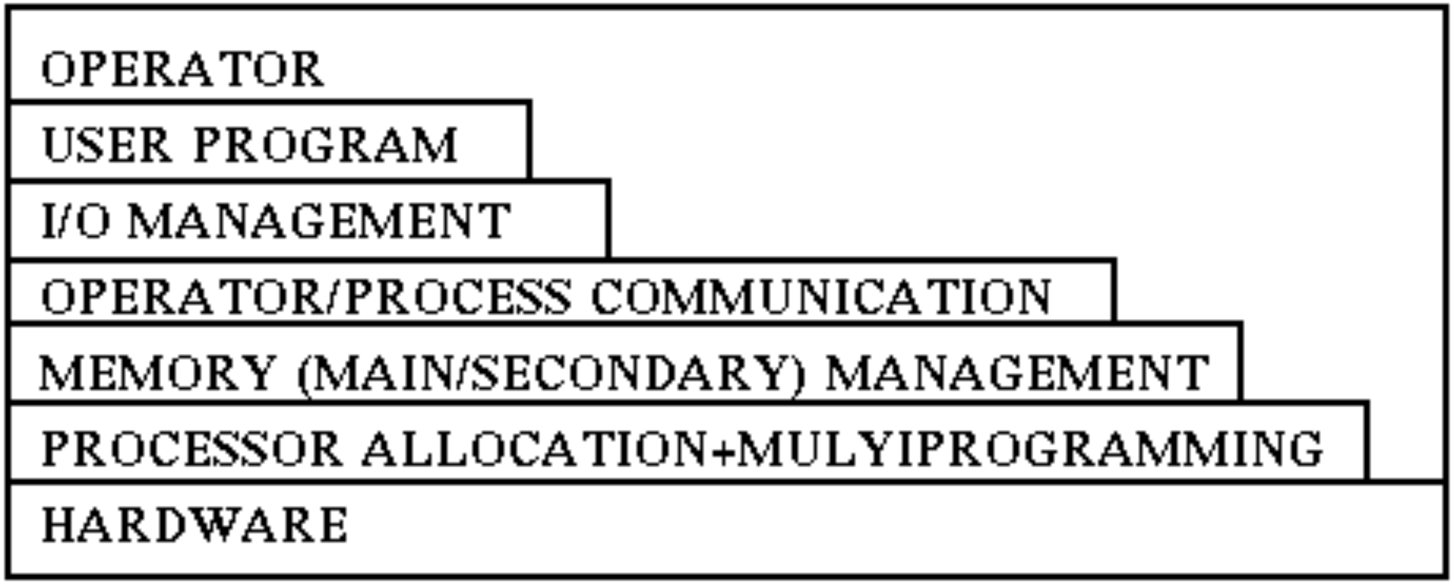


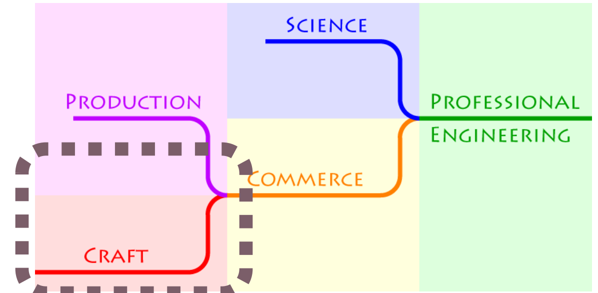
Fig. 3. Illustration of a property of separable programs.
A. A and B, linked as coroutines, communicate directly.
B. A writes its entire output before B reads anything.



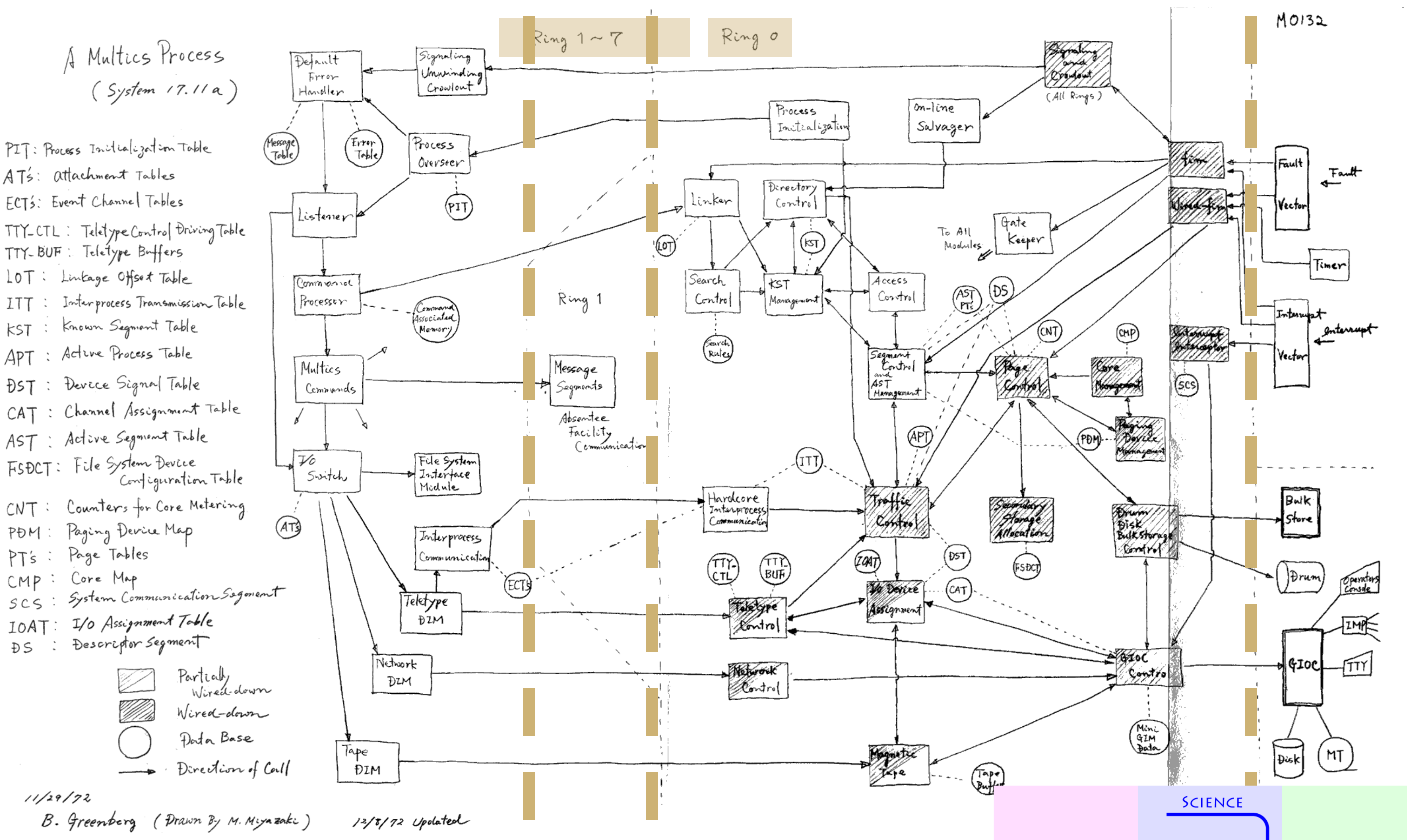
M. Conway: Design of a Separable Transition-diagram Compiler, CACM Jul 1963



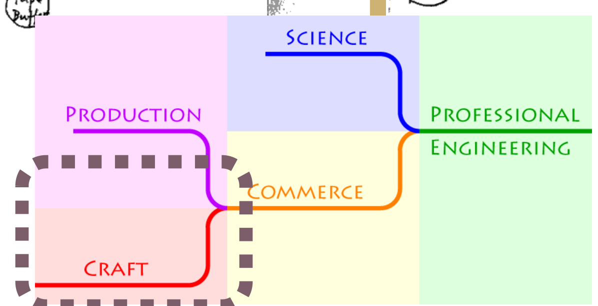
LAYERED SYSTEM (THE System, Dijkstra)



E.W. Dijkstra, The Structure of the “THE” Multiprogramming System. CACM May 1968



A layered system !!



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

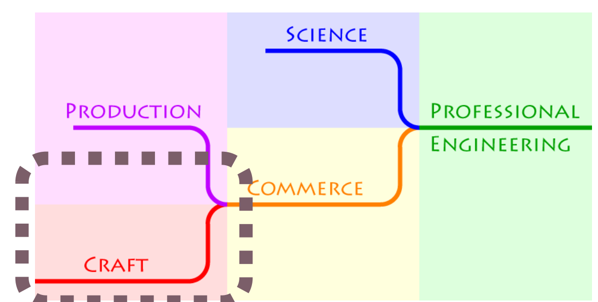
Craft practice

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



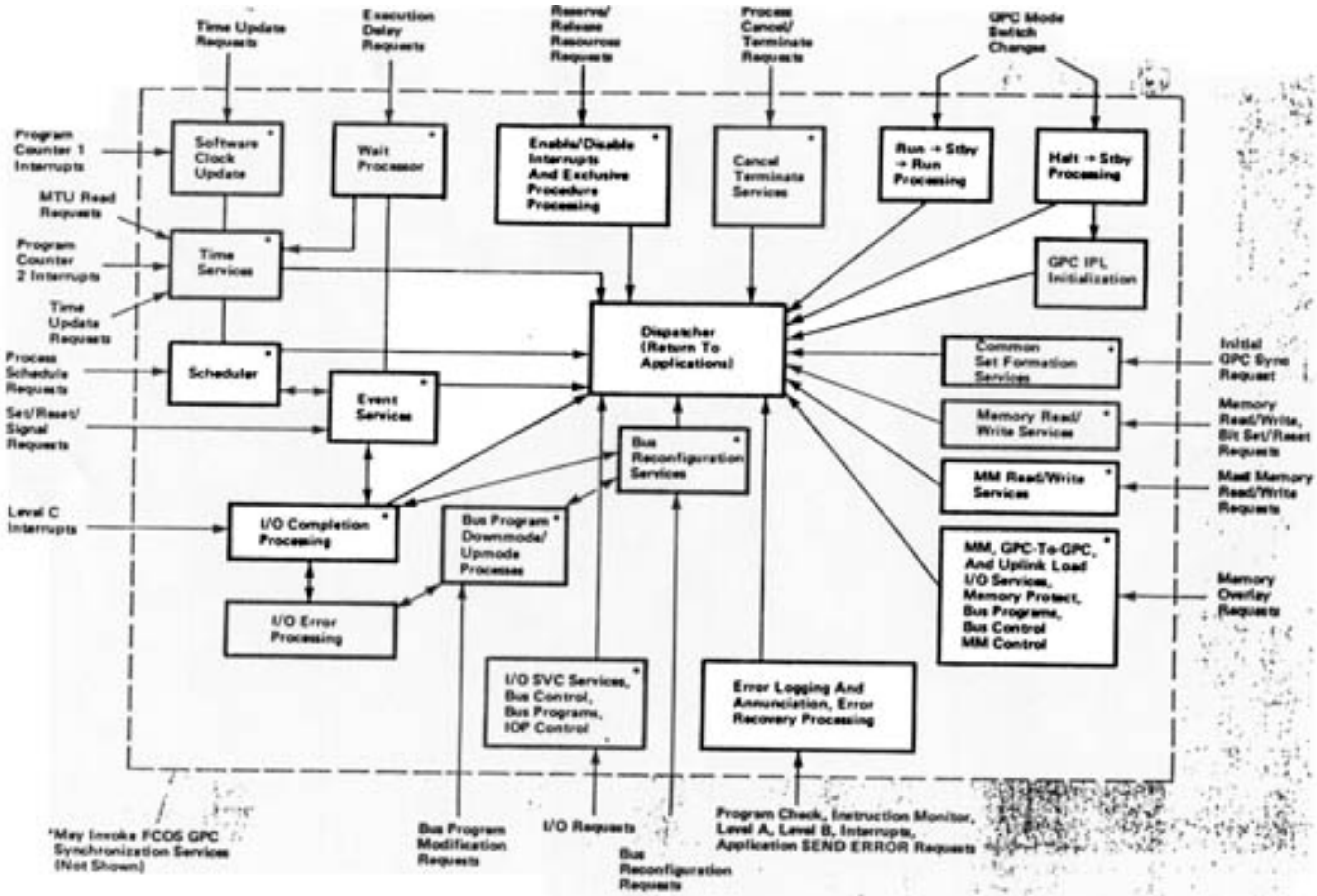
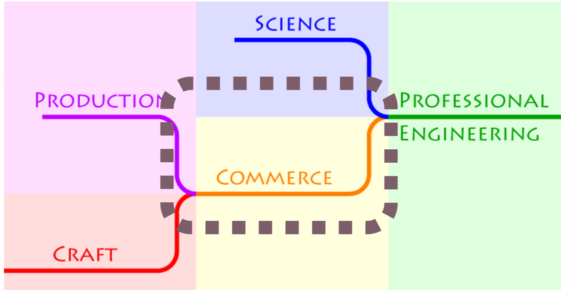


FIGURE 7. Flight Computer Operating System (The FCOS dispatcher coordinates and controls all work performed by the on-board computers.)

Communications of the ACM, "Architecture of the Space Shuttle Primary Avionics Software
September 1984, Vol. 27, No. 9, P. 933



Client Layer*

Access domain management
Buffering and record-level I/O
Transaction coordination

Agent Layer

Implementation of standard server interface
Logger, agent, and instance tasks

Helix Directories

Path name to FID mapping
Single-file (database) update by one task
Procedural interface for queries

Object (FID directory)

Identification and capability access (via FIDs)
FID to tree-root mapping; table of (FID,root,ref__count)
Existence and deletion (reference counts)
Concurrency control (file interlocking)

Secure Tree

Basic crash-resistant file structure
Conditional commit
Provision of secure array of blocks

System

Commit and restart authority
Disk space allocation
Commit domains

Cache

Caching and performance optimization
Commit support (flush)
Frame allocation (to domains)
Optional disk shadowing

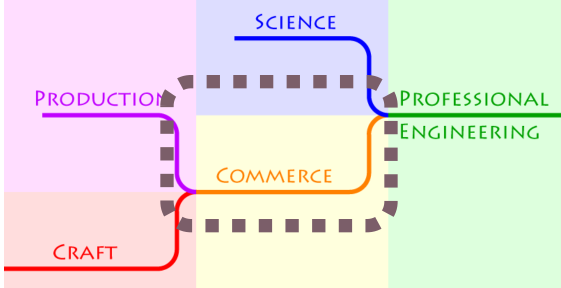
Canonical Disk

Physical disk access

*Also called client Helix.

Figure 2. Abstraction layering.

IEEE Software, "Helix: The architecture of the XMS Distributed File System, Marek Fridrich and William Older, May 1985, Vol. 2, No. 3, P. 23



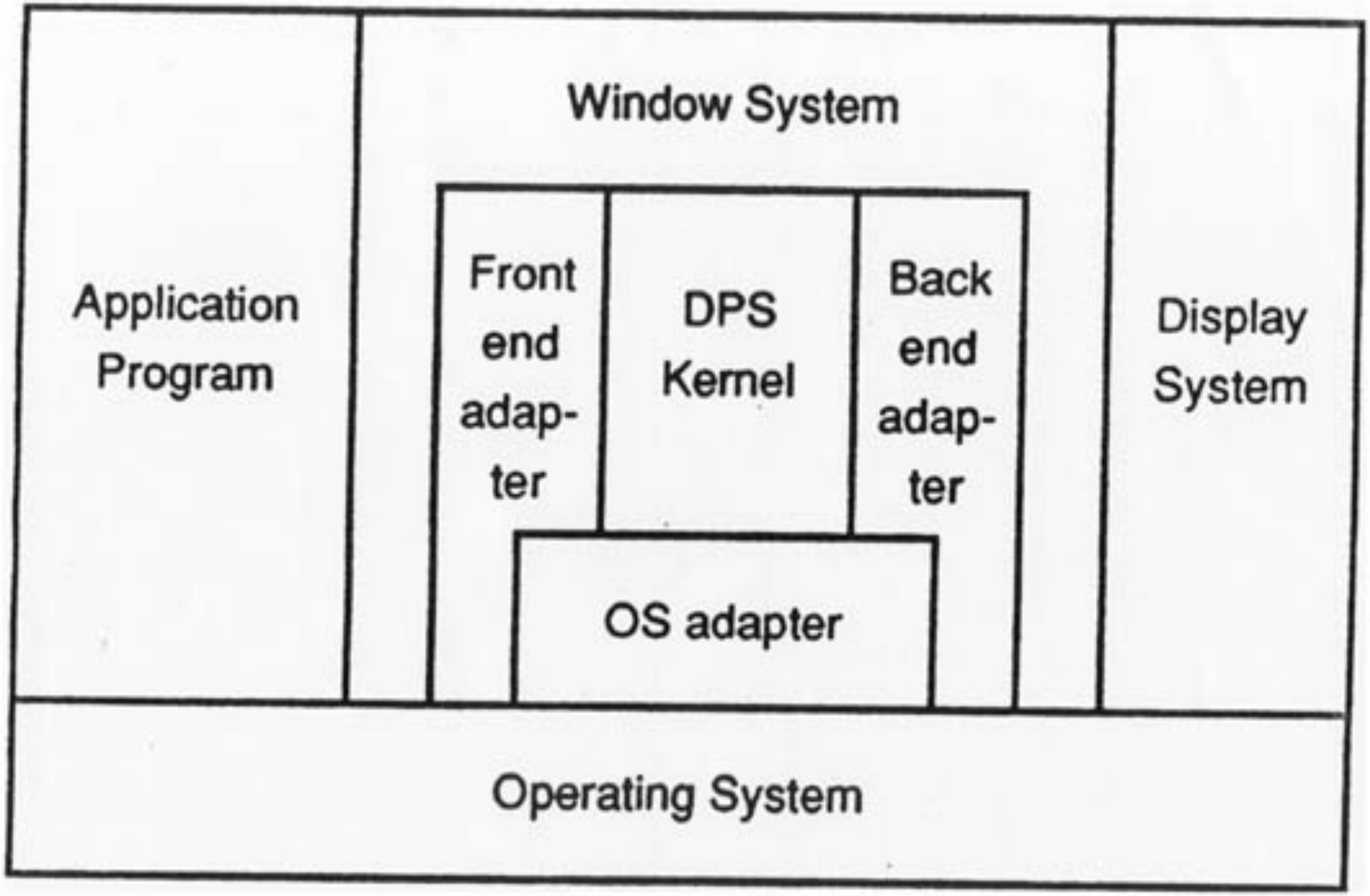
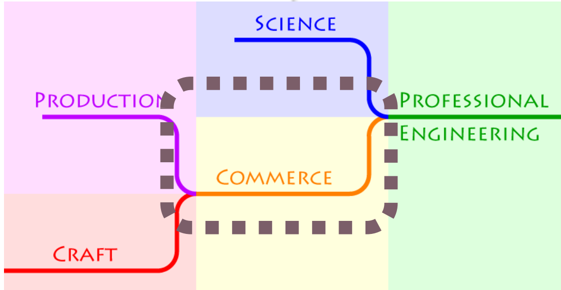
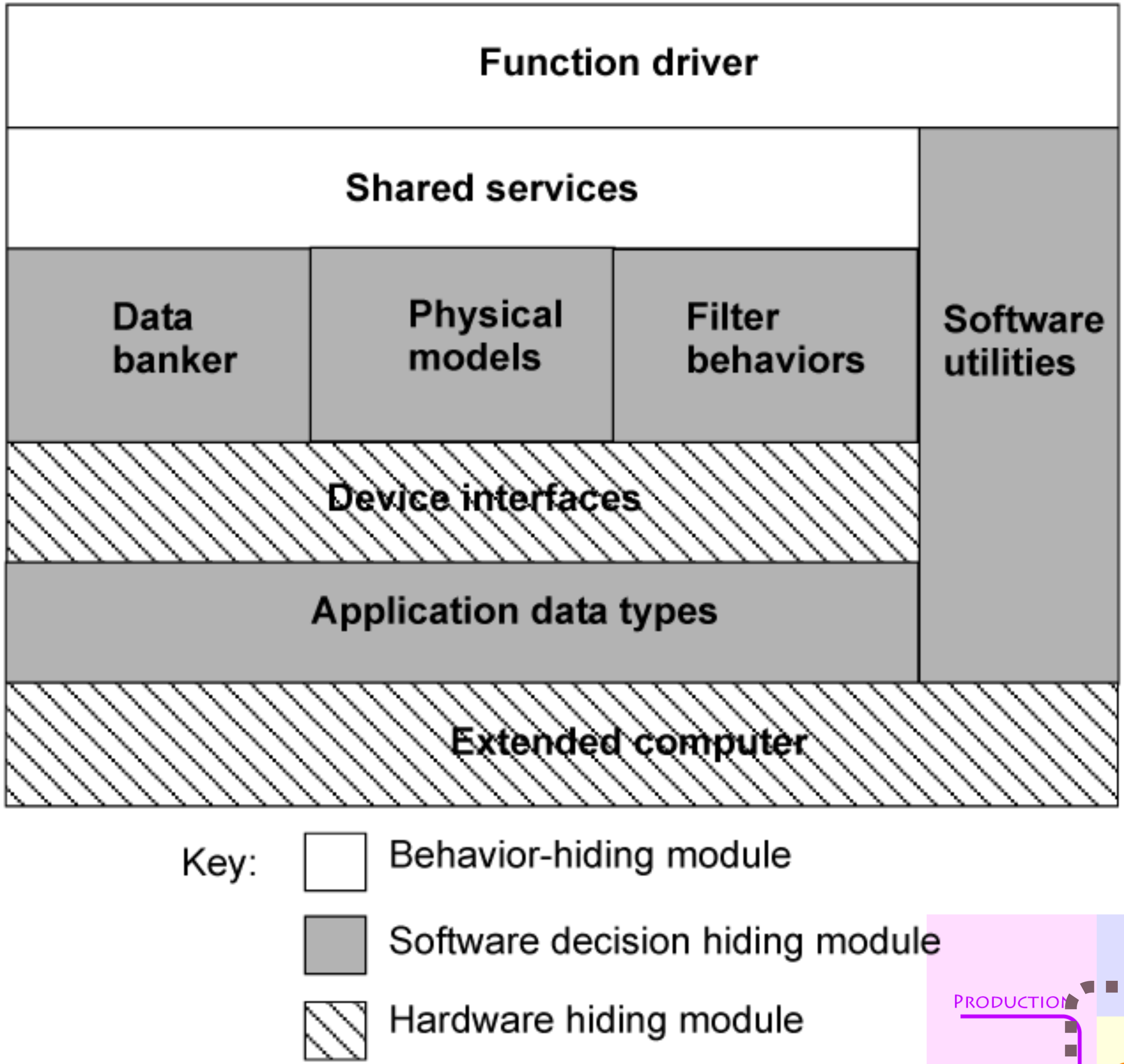
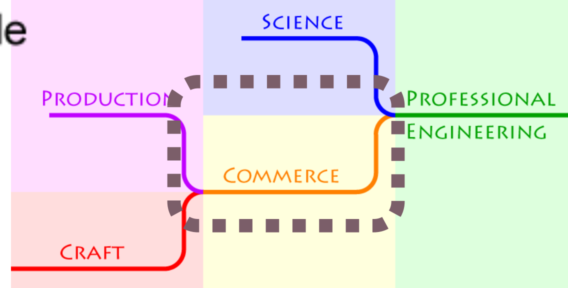


Figure 2. Display PostScript interpreter components.

An Overview of the DISPLAY POSTSCRIPT™ System, Adobe Systems Incorporated, March 16, 1988, P. 10



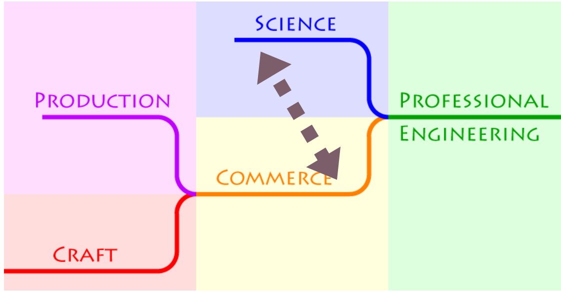




A7E avionics architecture, as shown in Bachman et al
Software Documentation in Practice, SEI 2000

Commerce stimulates science

uncertainty about quality, performance, changeability, etc	➔	models to support software metrics
ad hoc structure, multiple versions, interoperability issues, design drift	➔	styles /patterns for software architecture

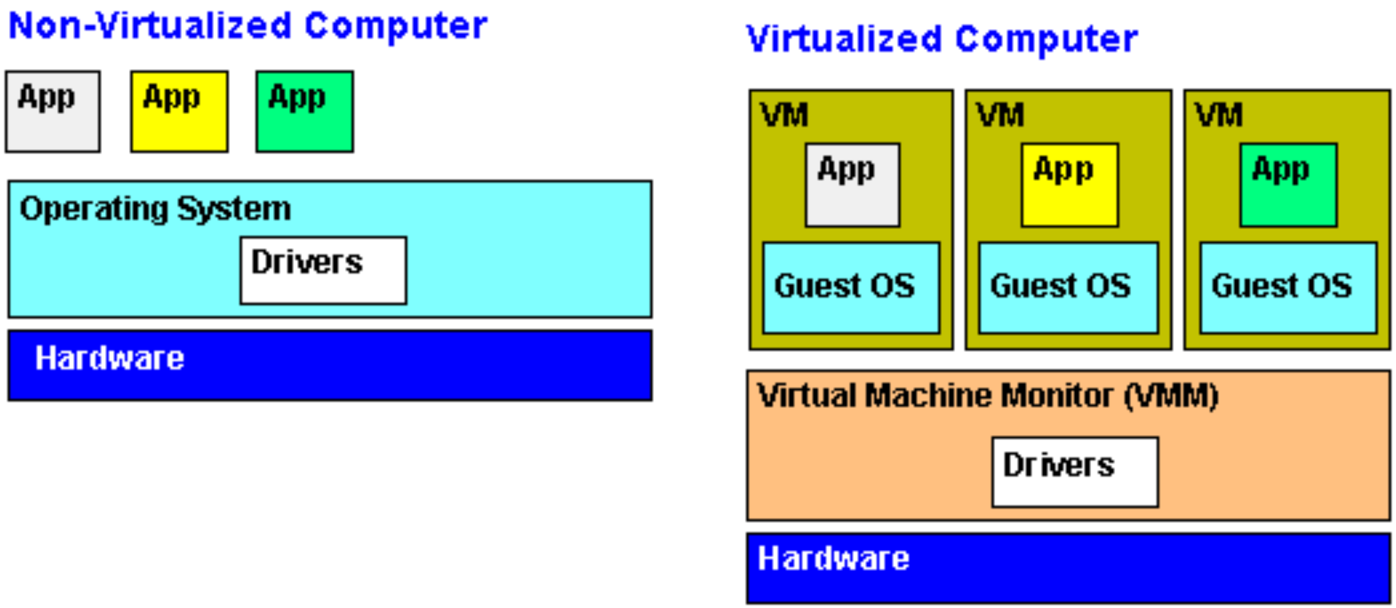


Sample idioms / styles / patterns

- 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>

Explanations for practitioners

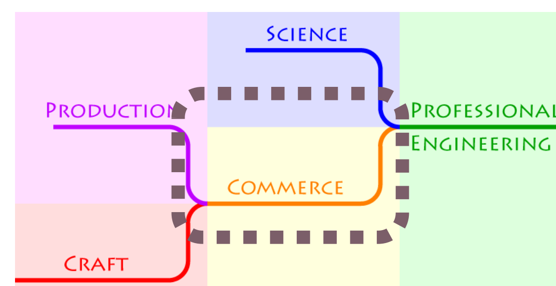
N-Tier architecture



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

Virtual machine

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



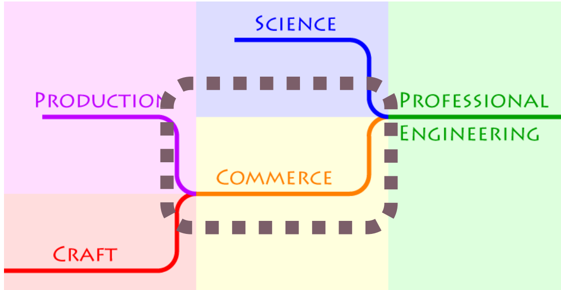
Commercial practice

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

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

1990s: early structure
- software product lines

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



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

Popular-ization
Propagate through community

Key Idea

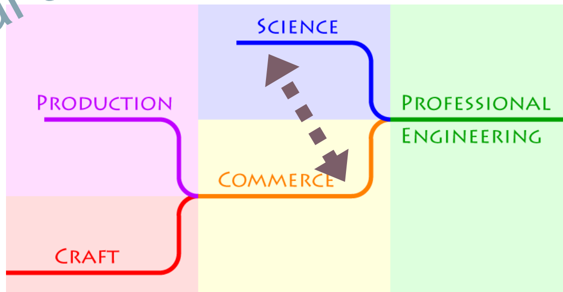
Seminal paper or system

Usable capability

Outsiders use it

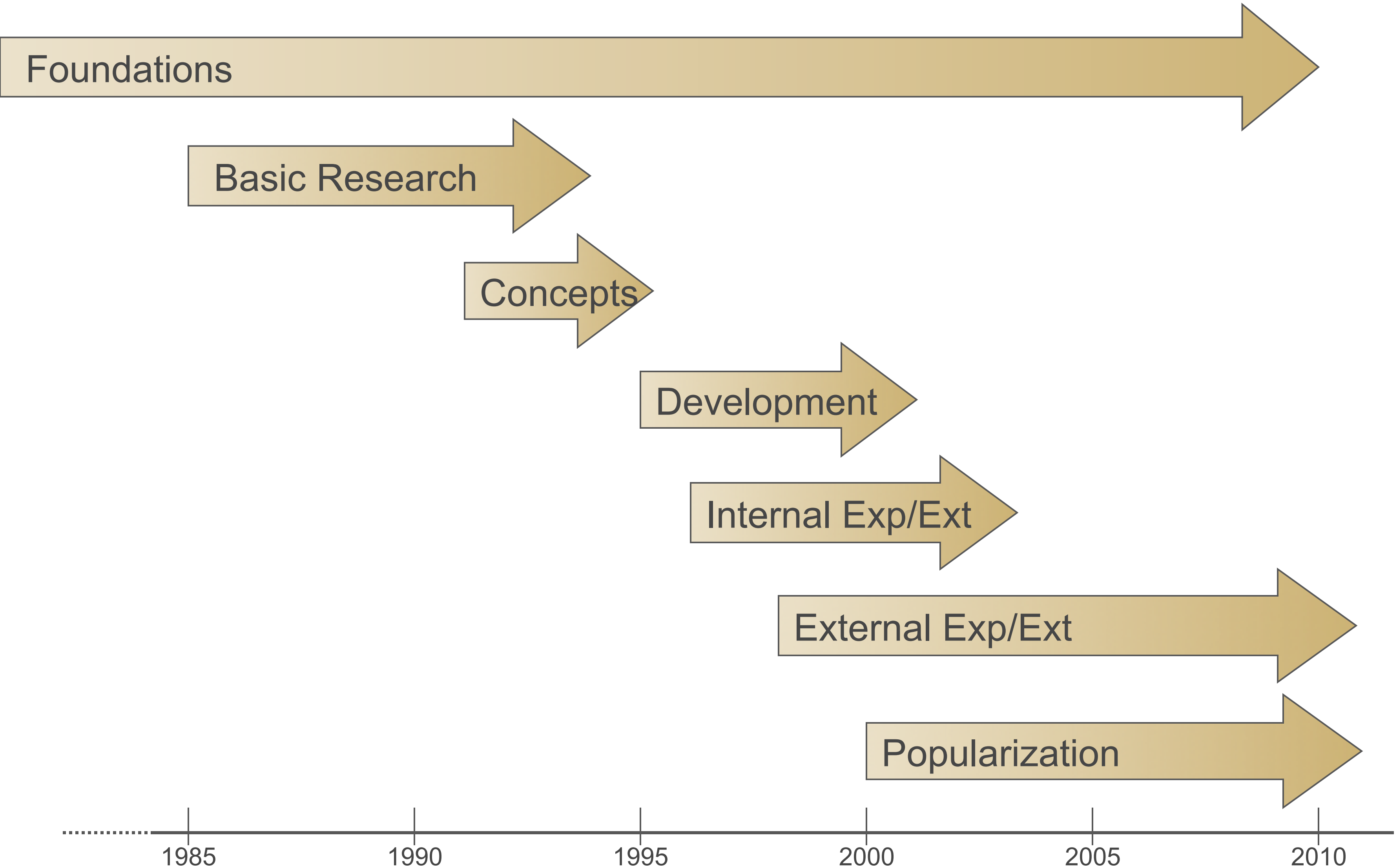
Production quality, commercial support

15-20 years



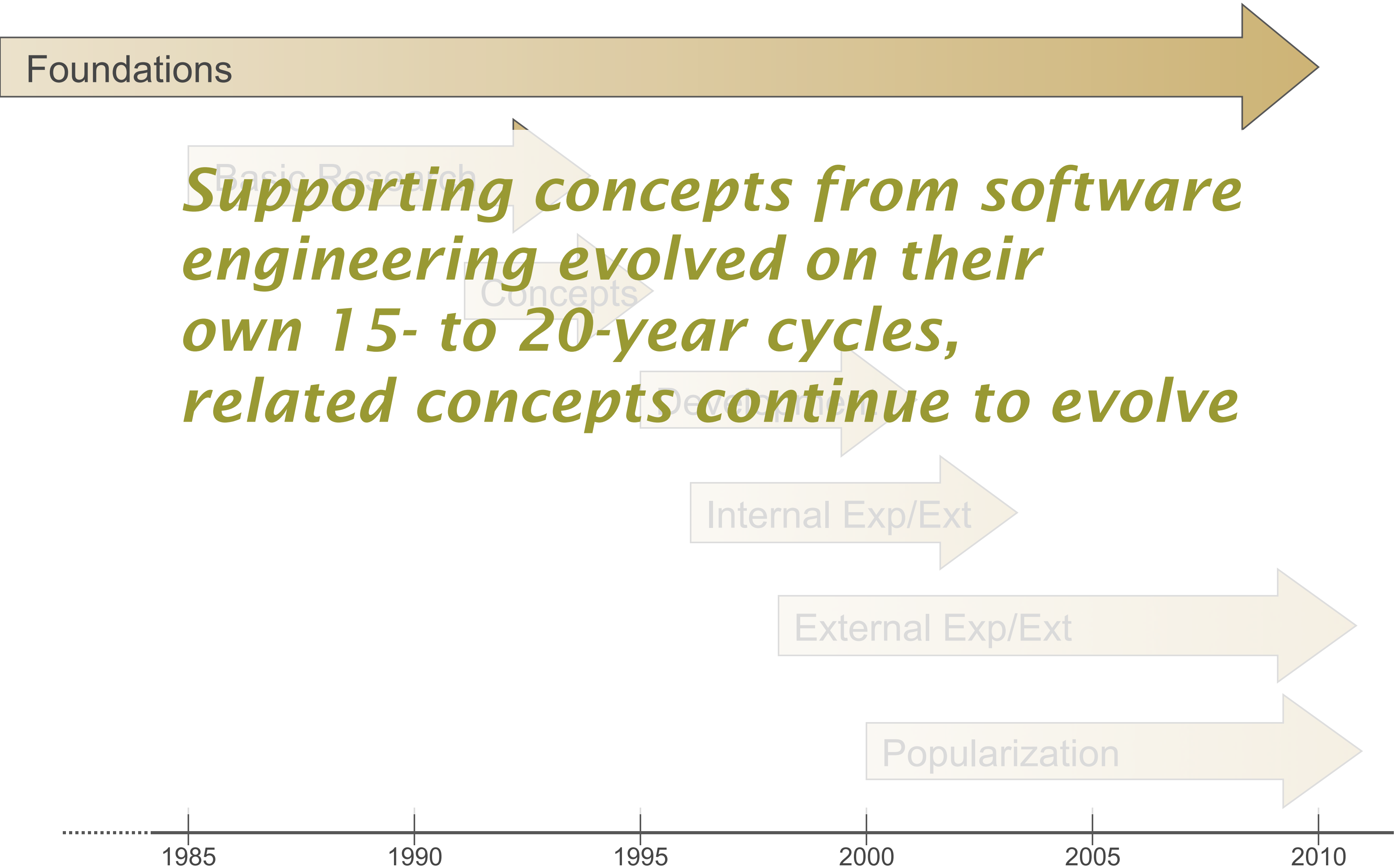
Sam Redwine, Jr. and William Riddle: *Software Technology Maturation*, Proc ICSE-8, May 1985

Maturation of software architecture



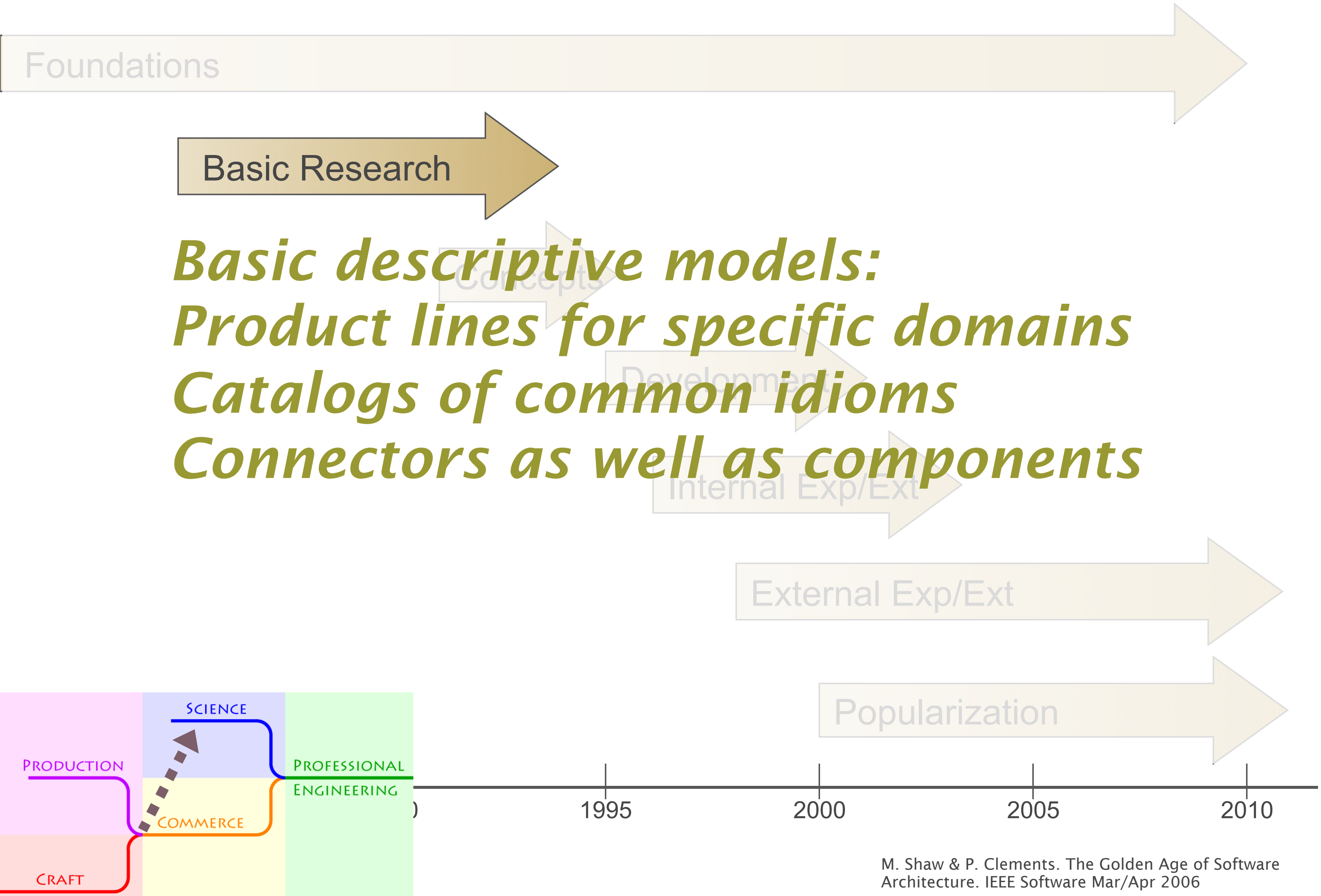
Garlan and Shaw. Software architecture: reflections on an evolving discipline. ESEC/FSE keynote 2011

Foundations



Supporting concepts from software engineering evolved on their own 15- to 20-year cycles, related concepts continue to evolve

Basic research, 1985-1993

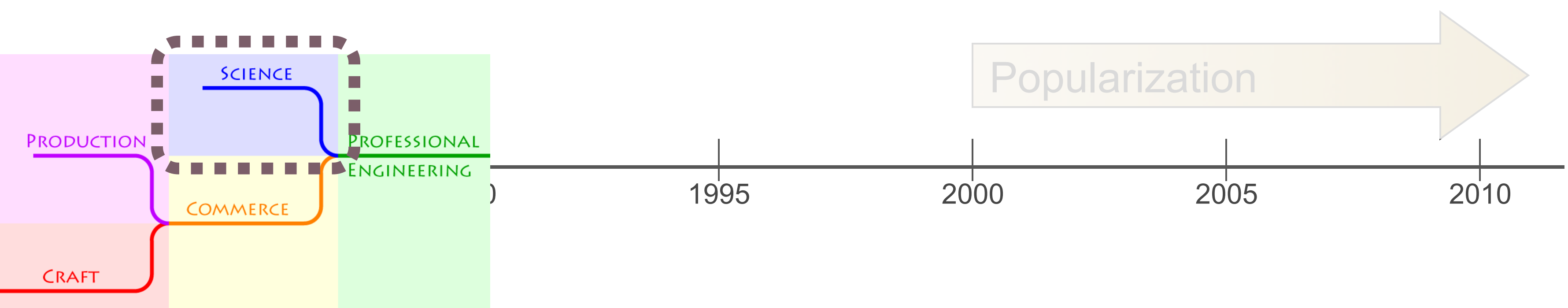


Basic descriptive models:
Product lines for specific domains
Catalogs of common idioms
Connectors as well as components

M. Shaw & P. Clements. The Golden Age of Software Architecture. IEEE Software Mar/Apr 2006

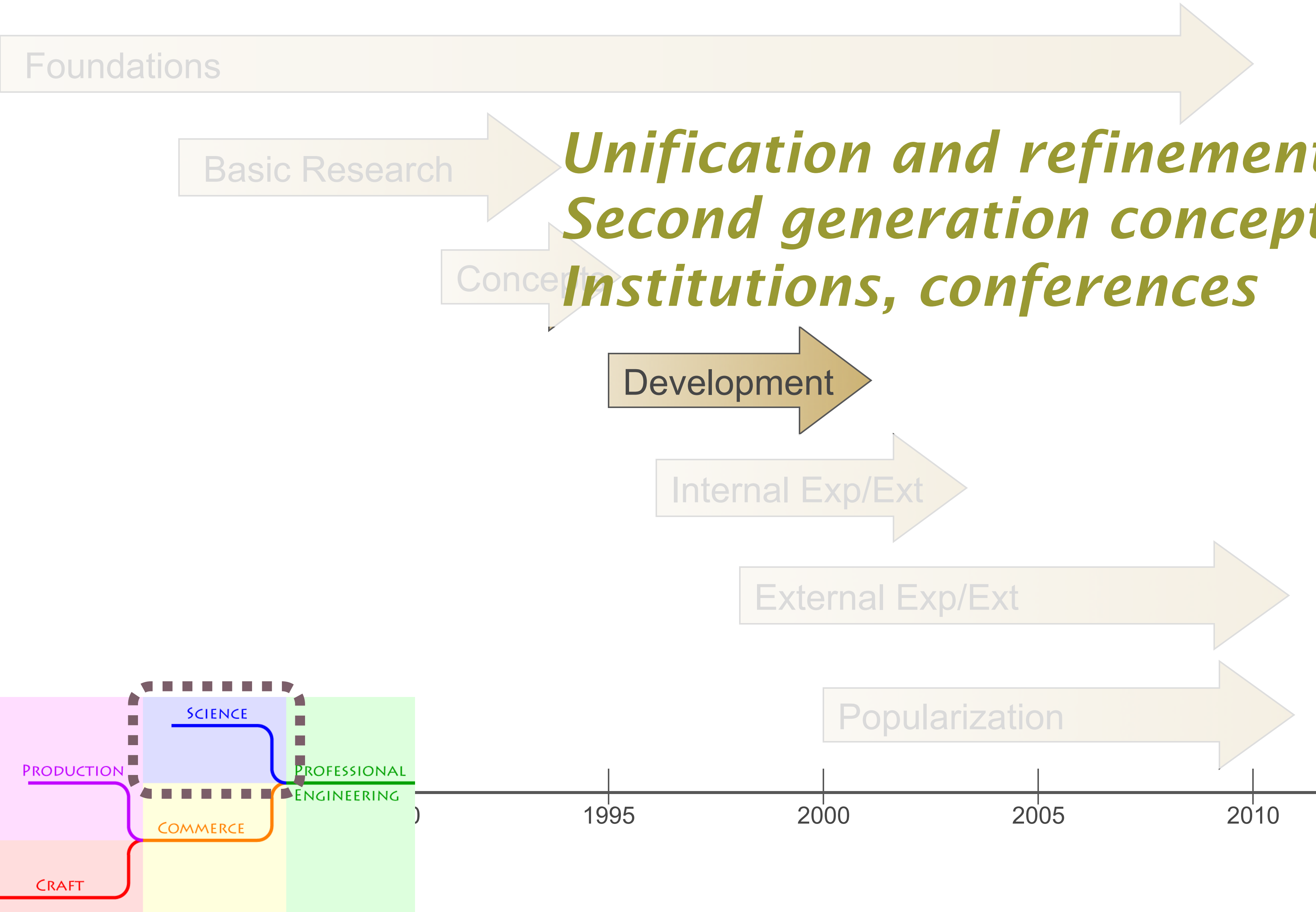
Concept formation 1992-1996

Elaboration of basic models
Languages and formalizations
Taxonomies of architectural patterns
Workshops and books



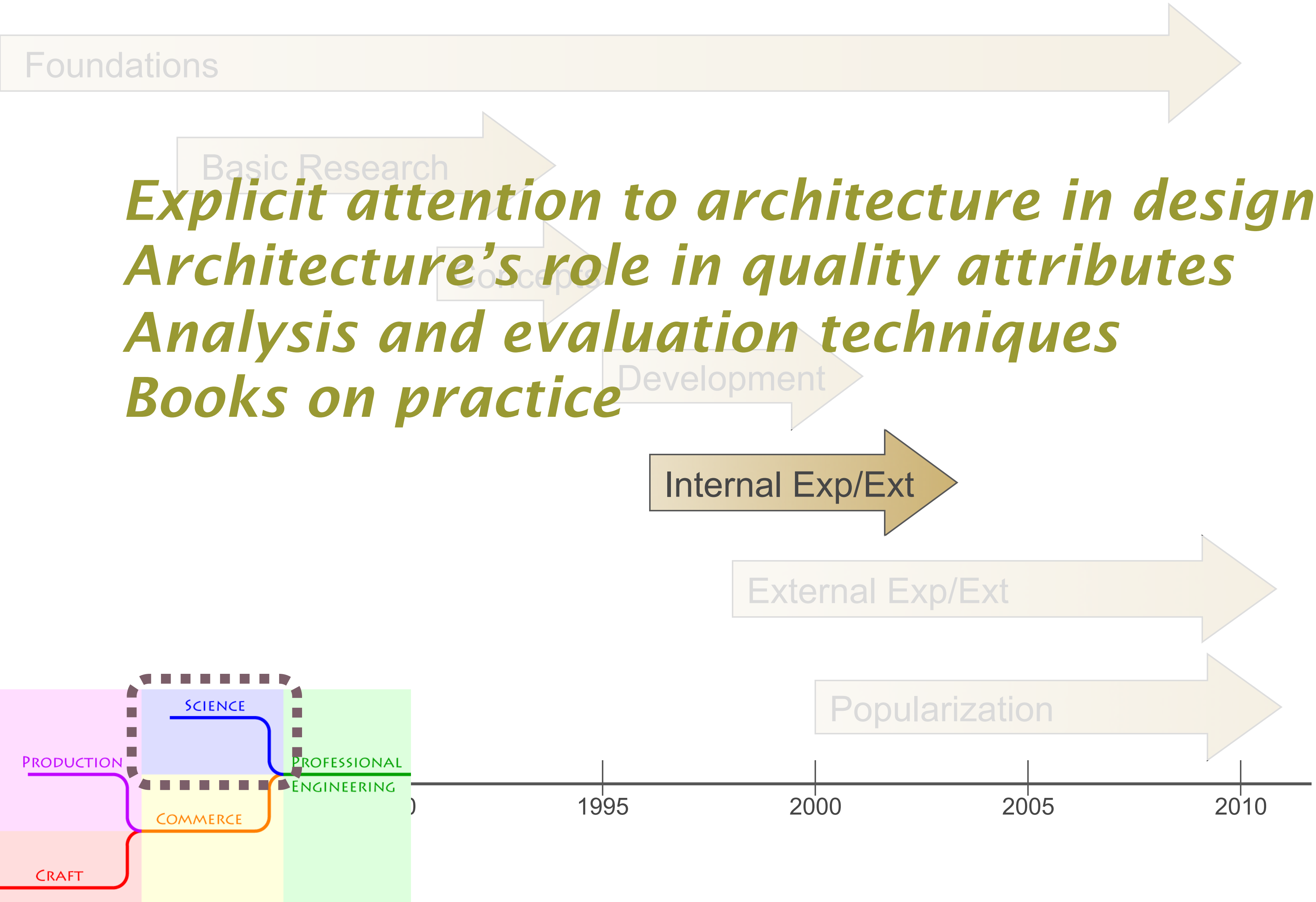
Development & extension: 1995-2000

Unification and refinement
Second generation concepts
Institutions, conferences



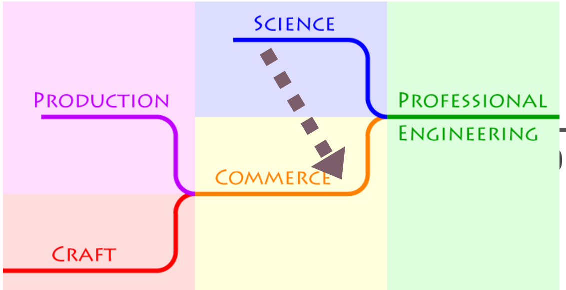
Internal exploration: 1996-2003

Explicit attention to architecture in design
Architecture's role in quality attributes
Analysis and evaluation techniques
Books on practice



External exploration: 1998-present

Technologies useful beyond development group
Tools and frameworks
Company-specific architectures

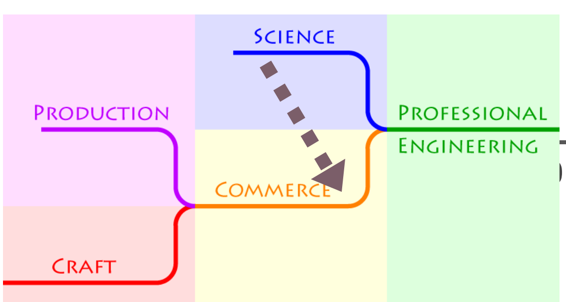


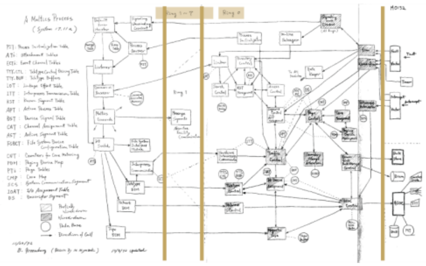
Popularization: 2000-present

Production-quality, supported, commercialized technology, standards

Education, professional organizations

Architect as senior technical leader





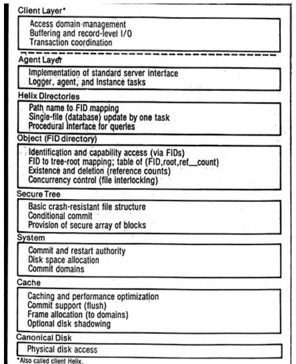
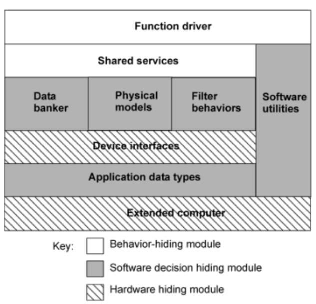
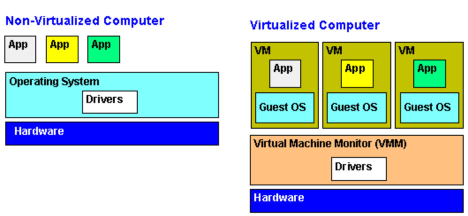


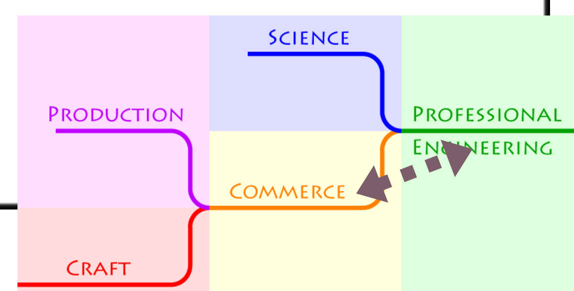
Figure 2. Abstraction layering.

IEEE Software, "Helix: The architecture of the XMS Distributed File System, Marek Fridrich and William Older, May 1985, Vol. 2, No. 3, P. 23



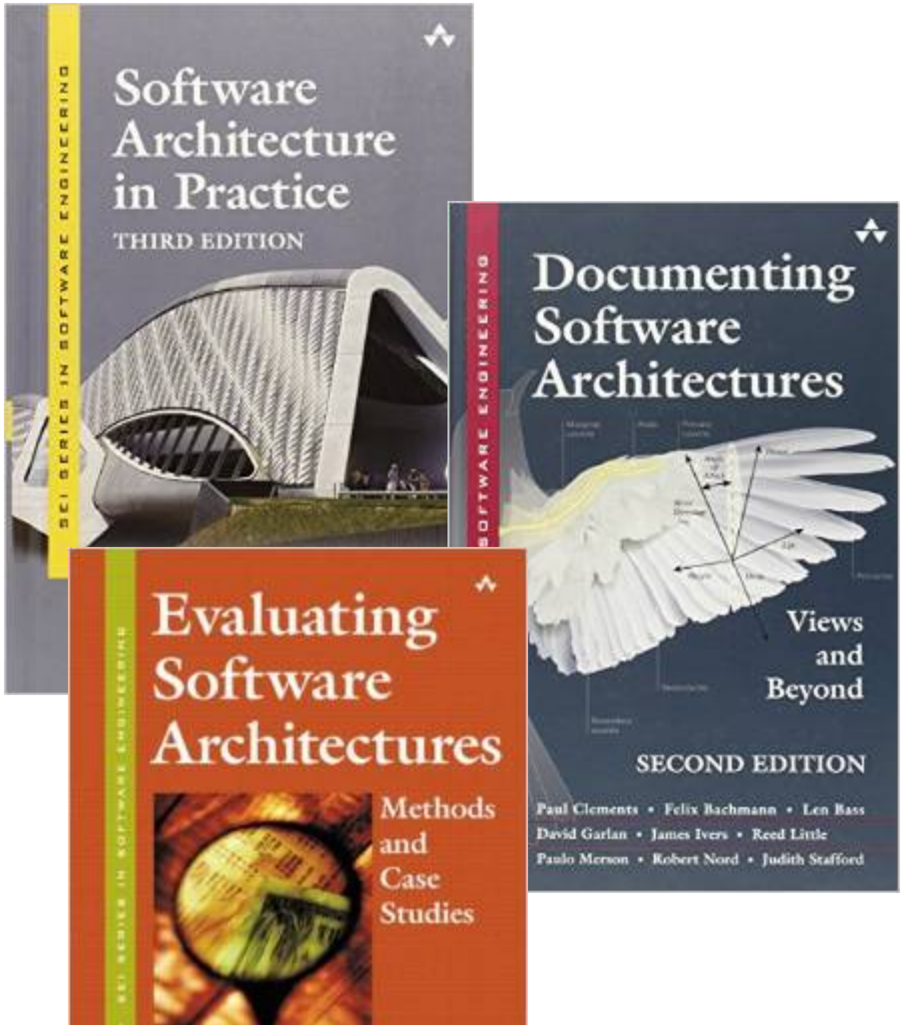


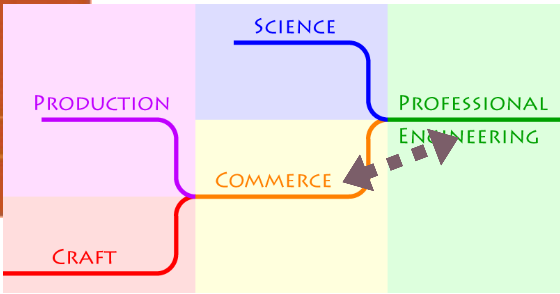
http://xkcd.com/676/



Systematically Organized Knowledge

SEI Series organizes knowledge about architecture and its analysis





Architectural styles and reasoning

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

Shaw, Clements. Toward Boxology. ISAW-2, 1996.

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.

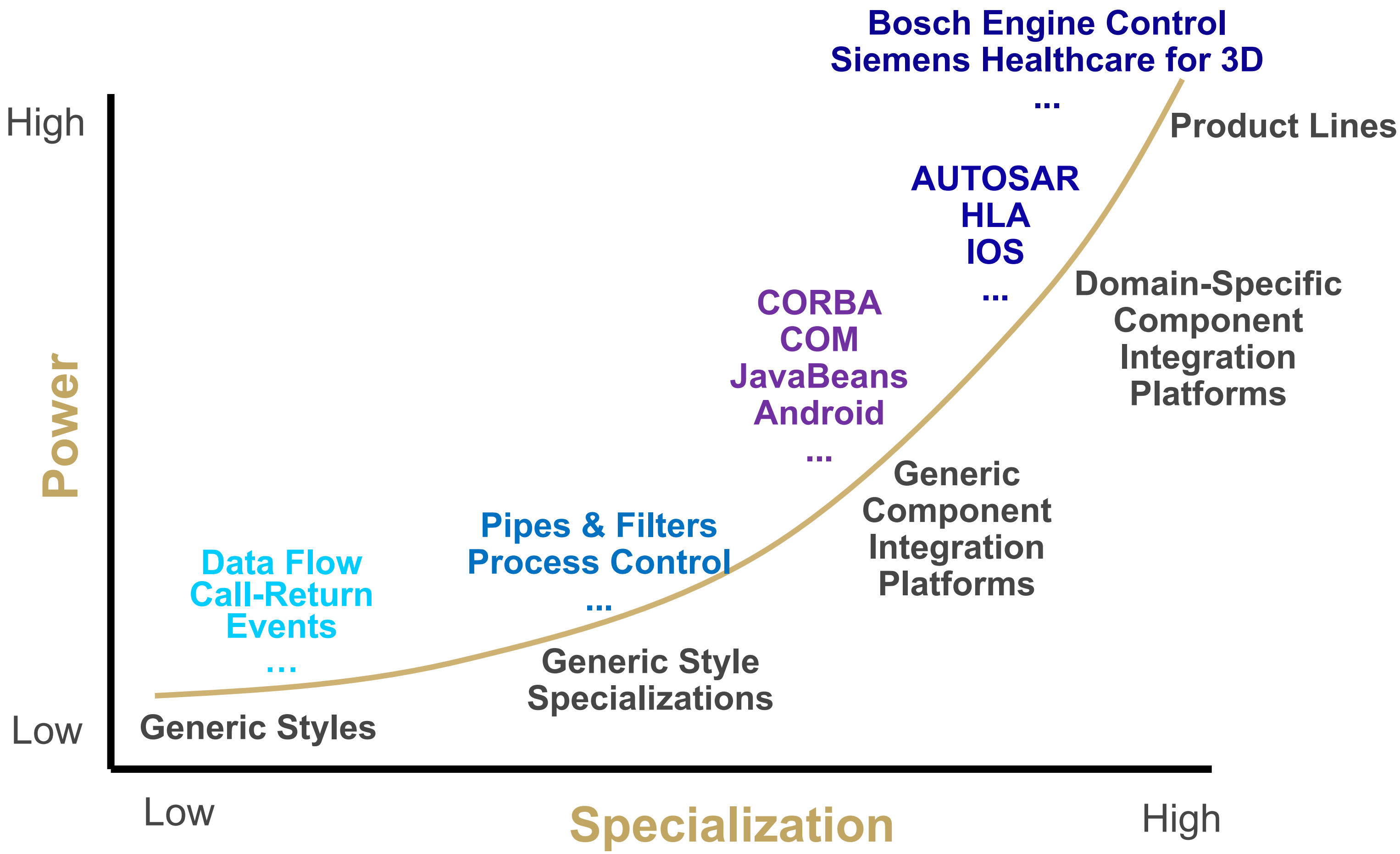
- However, if your problem involves passing rich data representations, avoid pipelines restricted to ASCII.

If your system involves controlling continuing action, is embedded in a physical system, and is subject to unpredictable external perturbation so that preset algorithms go awry, consider ***closed loop*** architectures.

Shaw, Clements. Toward Boxology. ISAW-2, 1996.

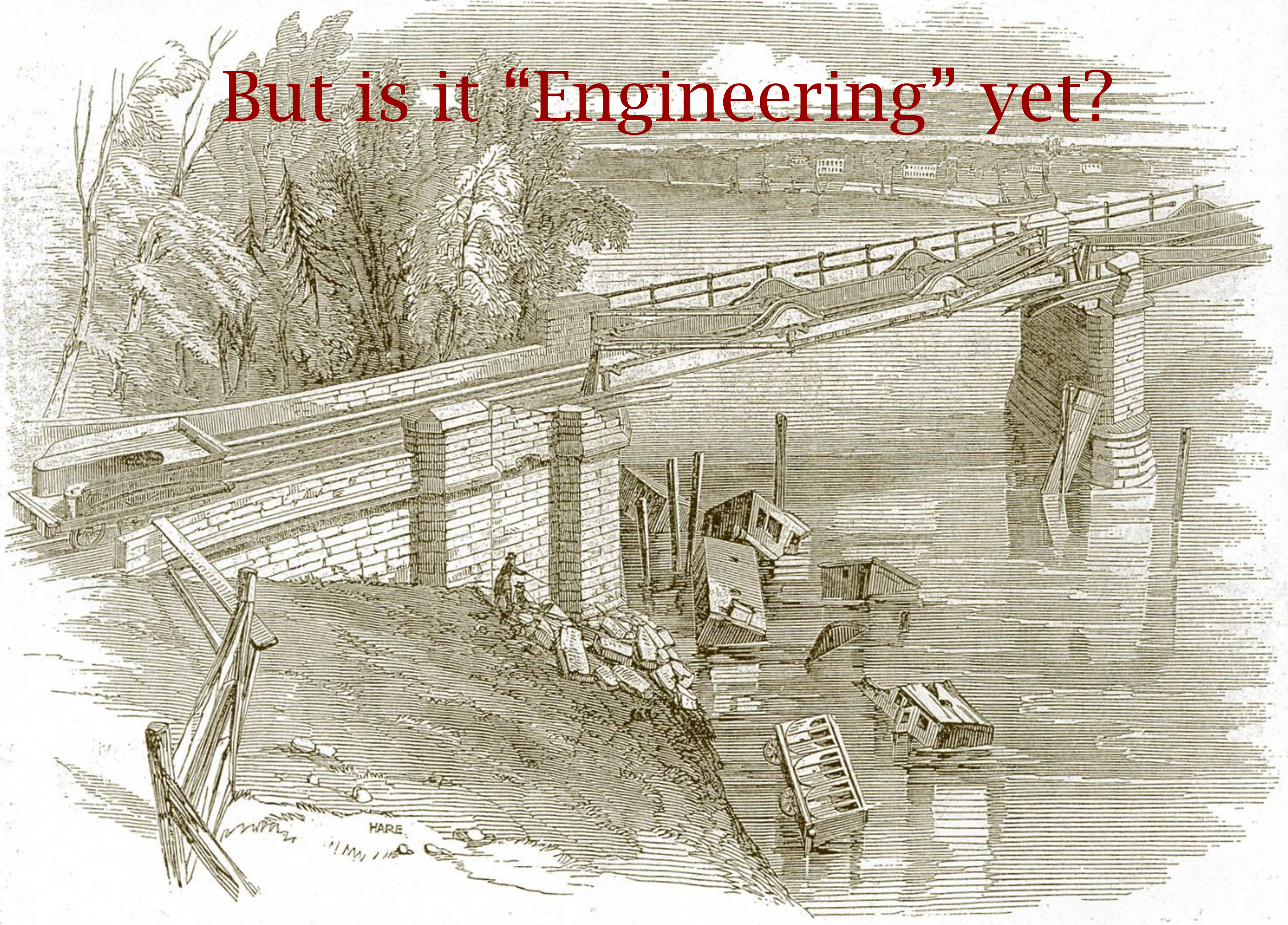
Generality-power trades

Styles, Platforms, and Product Lines



Garlan. Software Architecture: A Travelogue. ICSE 2014.

But is it “Engineering” yet?



SCENE OF THE LATE RAILWAY ACCIDENT, AT CHESTER.—DILAPIDATED SPAN OF THE DEE BRIDGE.

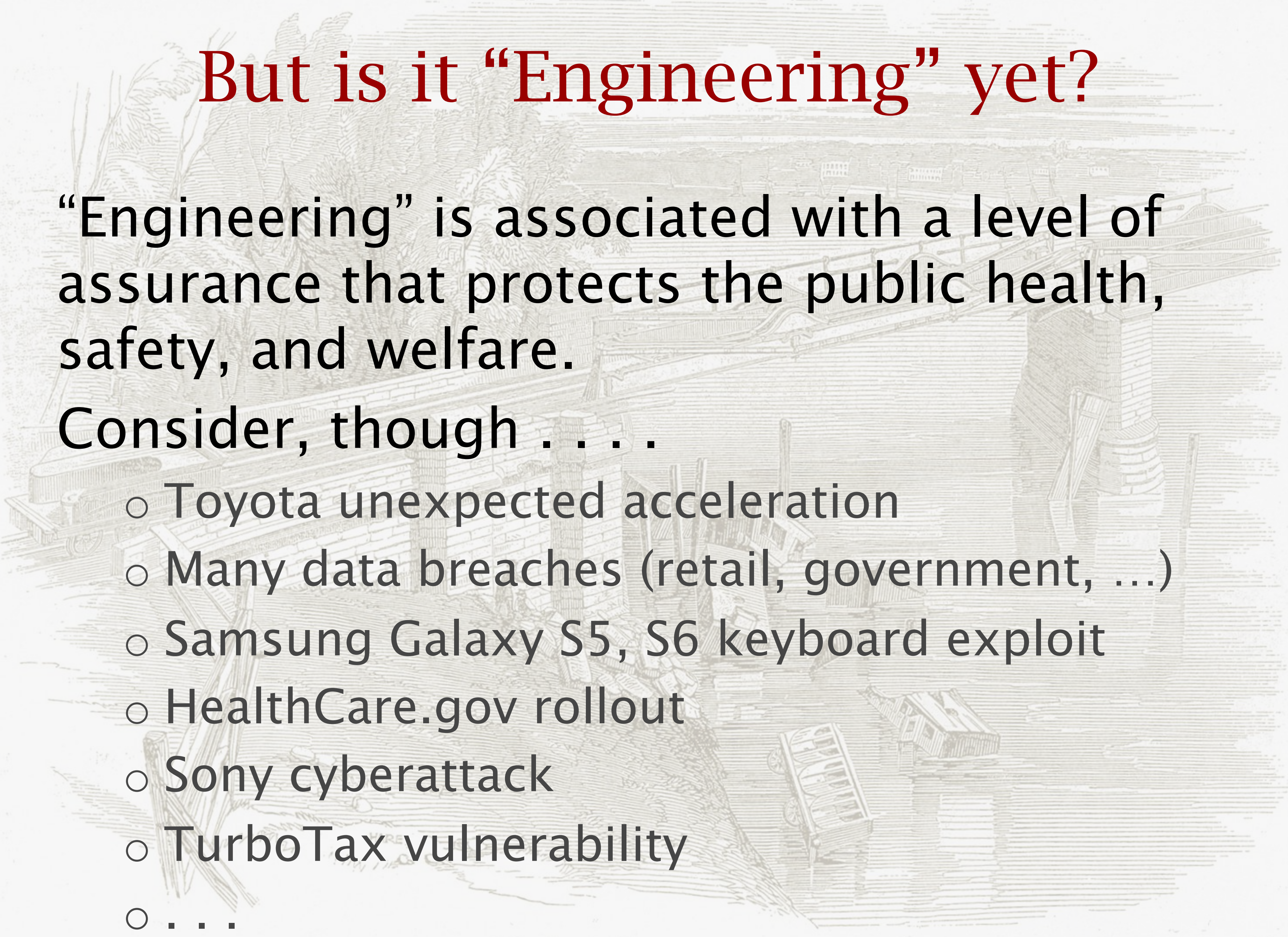
Illustrated London News, 1847

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
- Many data breaches (retail, government, …)
- Samsung Galaxy S5, S6 keyboard exploit
- HealthCare.gov rollout
- Sony cyberattack
- TurboTax vulnerability
- . . .



SCENE OF THE LATE RAILWAY ACCIDENT, AT CHESTER.—DILAPIDATED SPAN OF THE DEE BRIDGE.

Illustrated London News, 1847

Toyota unintended acceleration

- Throttle stuck open, driver couldn’t stop car
  - Hundreds died/injured in 2002-2010 models
  - Toyota denied claims but settled for $1.6++ Billion
- Electronic Throttle Control System (ETCS)
  - wide open throttle → brakes won’t stop car
  - single-bit failure could kill critical subtask
- Software didn’t follow known good practices
  - watchdog didn’t detect major task failure
  - cyclomatic complexity often over 50
  - poor coding practice, ~10,000 global variables
  - recursion could cause uncaught stack overflow
  - poor development/testing process compliance

Phil Koopman http://betterembsw.blogspot.com/2014/09/a-case-study-of-toyota-unintended.html

Identity Theft Resource Center

2014 Data Breach Category Summary

How is this report produced? What are the rules? See last page of report for details. Report Date: 1/5/2015

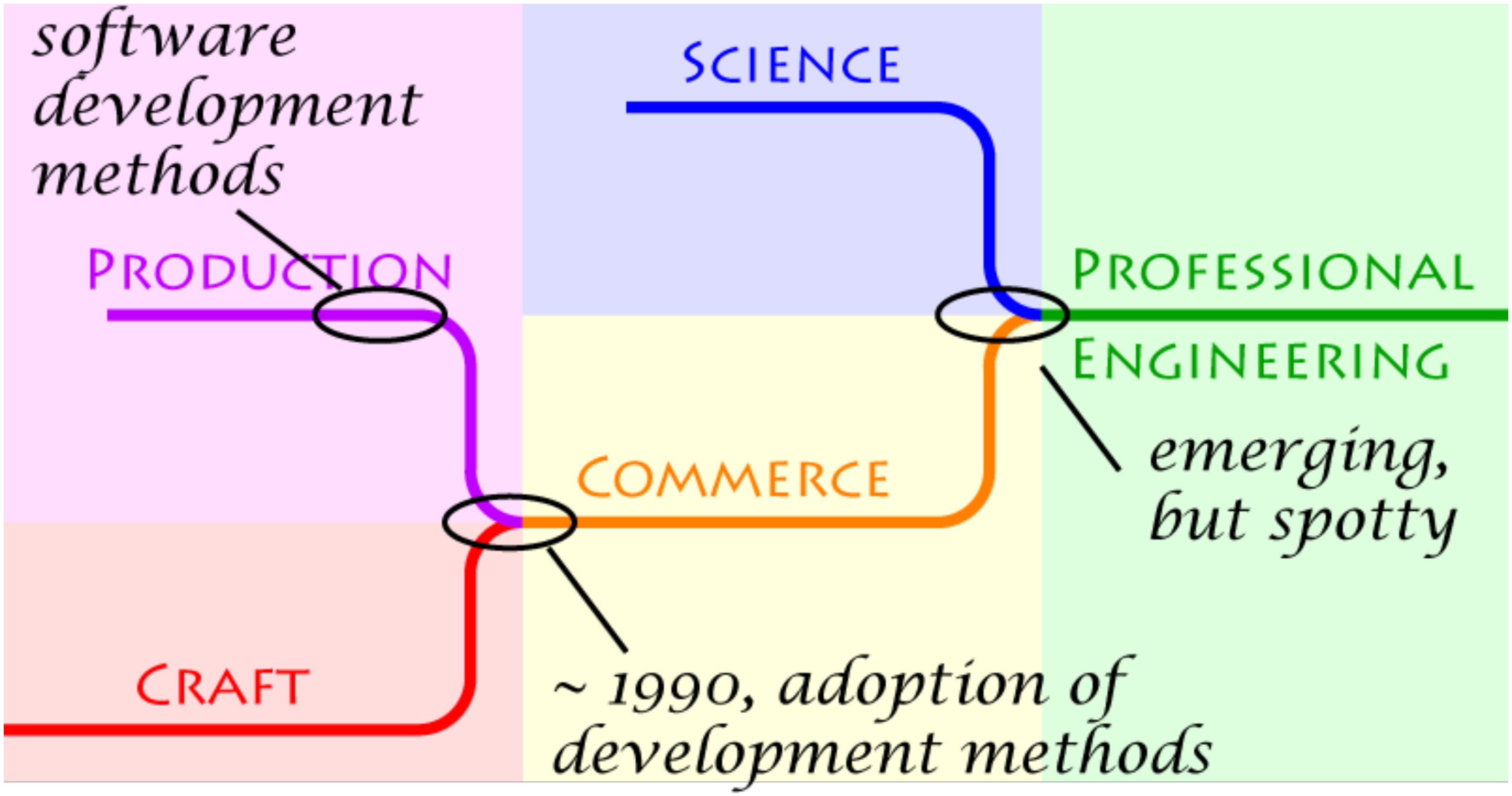
Totals for Category:	**Banking/Credit/Financial**	**# of Breaches: 43**	**# of Records:**	**1,198,492**
		% of Breaches: 5.5%	**%of Records:**	**1.4%**
Totals for Category:	**Business**	**# of Breaches: 258**	**# of Records:**	**68,237,914**
		% of Breaches: 33.0	**%of Records:**	**79.7%**
Totals for Category:	**Educational**	**# of Breaches: 57**	**# of Records:**	**1,247,812**
		% of Breaches: 7.3%	**%of Records:**	**1.5%**
Totals for Category:	**Government/Military**	**# of Breaches: 92**	**# of Records:**	**6,649,319**
		% of Breaches: 11.7	**%of Records:**	**7.8%**
Totals for Category:	**Medical/Healthcare**	**# of Breaches: 333**	**# of Records:**	**8,277,991**
		% of Breaches: 42.5	**%of Records:**	**9.7%**
Totals for All Categories:		**# of Breaches: 783**	**# of Records:**	**85,611,528**
		% of Breaches: 100.0	**%of Records:**	**100.0%**

2014 Breaches Identified by the ITRC as of: 1/5/2015

Total Breaches: 783
Records Exposed: 85,611,528

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

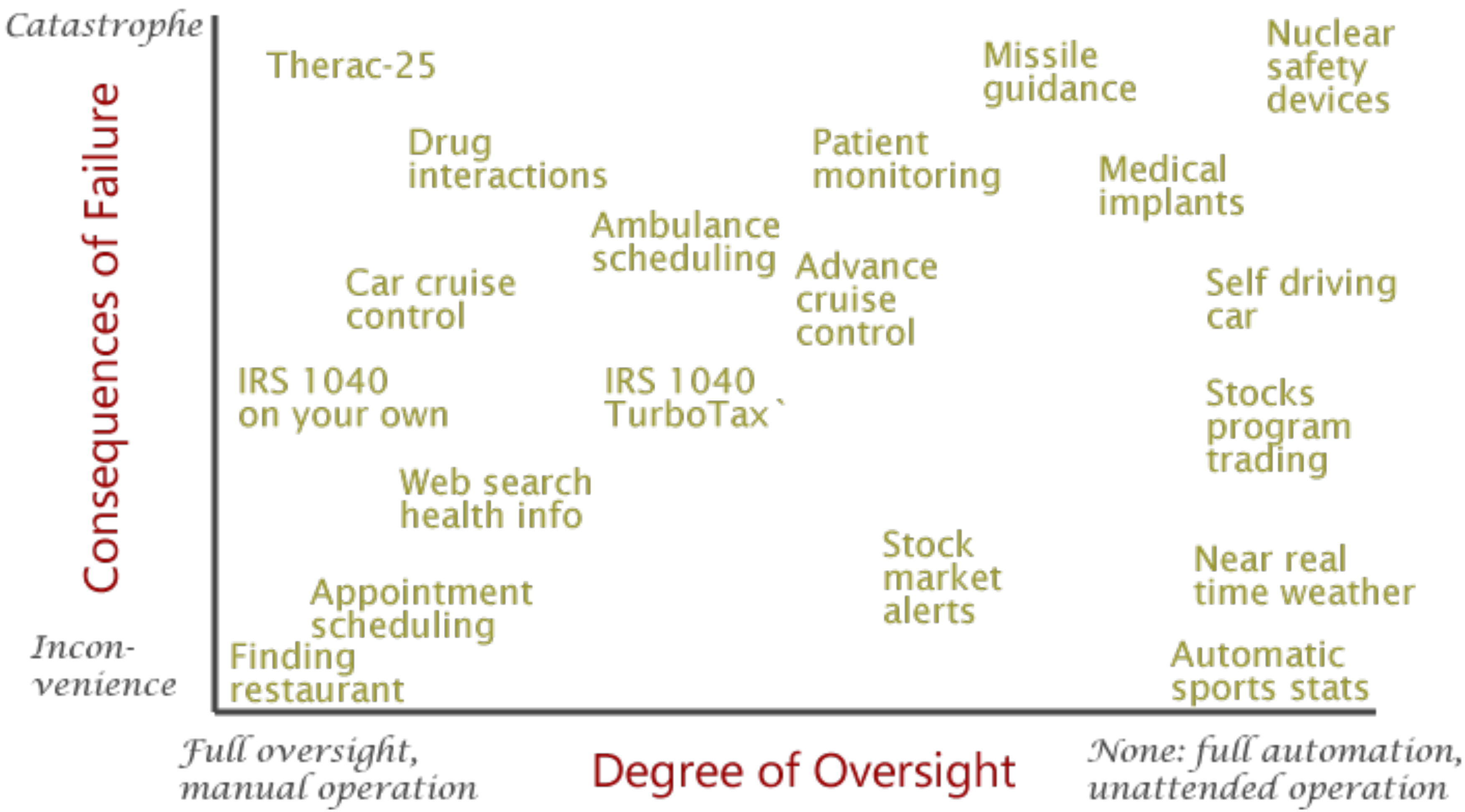


software development methods
Production
Science
Professional
Engineering
emerging, but spotty
Commerce
Craft
~ 1990, adoption of development methods

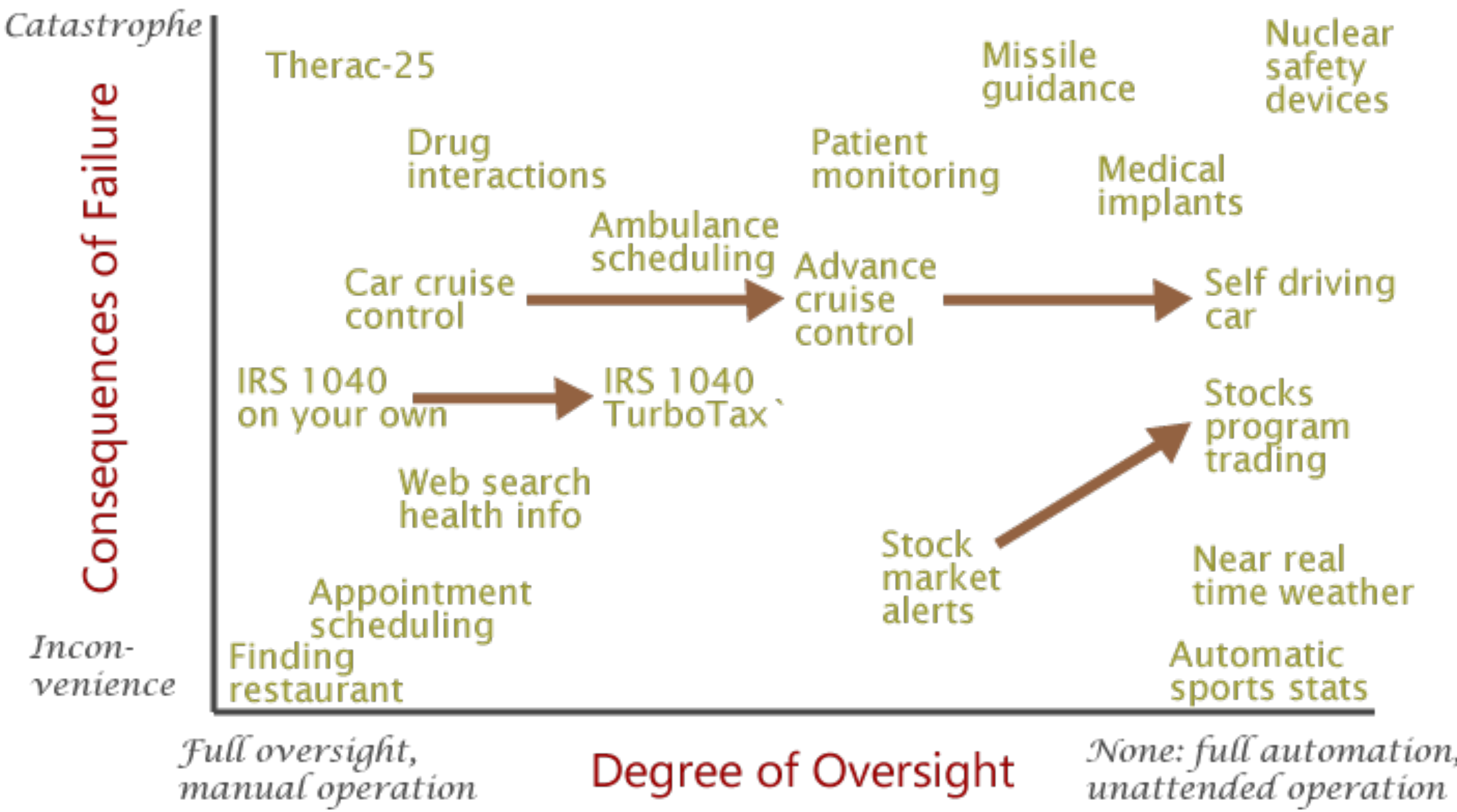
Want to be part of this?

http://isri.cmu.edu/education/

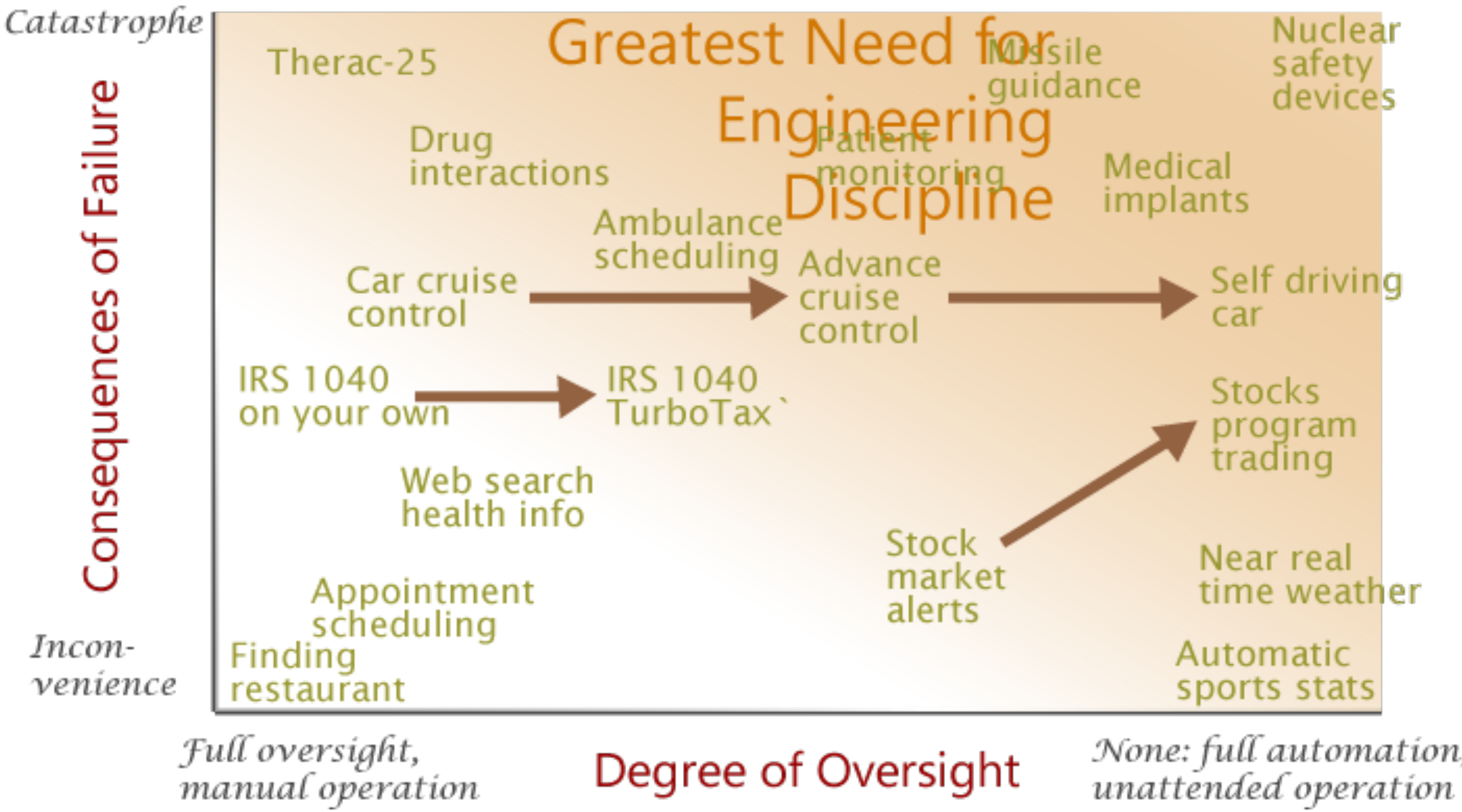
Making Progress



Shaw: Sufficient Correctness and Homeostasis in Open Resource Coalitions: How Much Can You Trust Your Software System? ISAW-4, 2000



Shaw: Sufficient Correctness and Homeostasis in Open Resource Coalitions: How Much Can You Trust Your Software System? ISAW-4, 2000



Catastrophe
Incon-
venience
Consequences of Failure
Greatest Need for Engineering Discipline
Therac-25
Missile guidance
Nuclear safety devices
Drug interactions
Patient monitoring
Medical implants
Ambulance scheduling
Car cruise control
Advance cruise control
Self driving car
IRS 1040 on your own
IRS 1040 TurboTax`
Stocks program trading
Web search health info
Stock market alerts
Near real time weather
Appointment scheduling
Finding restaurant
Automatic sports stats
Full oversight, manual operation
Degree of Oversight
None: full automation, unattended operation

Adapting to evolving technology

- Technology outruns traditional manuals
  - Understand how search supplants indexing
  - Analog of MapReduce for documentation?
- Agility, "perpetual beta", and evolution
  - Exploit power end of generality tradeoff, embedding knowledge in task-specific tools
- Scaling cost to consequence, predictably
  - High stakes applications have rigorous engineering, mashups are fine for throwaways – but where is middle ground?
- *How do we bring codified knowledge to design? Exhortation won't work*

Civilize the electronic frontier

Infrastructure and amenities

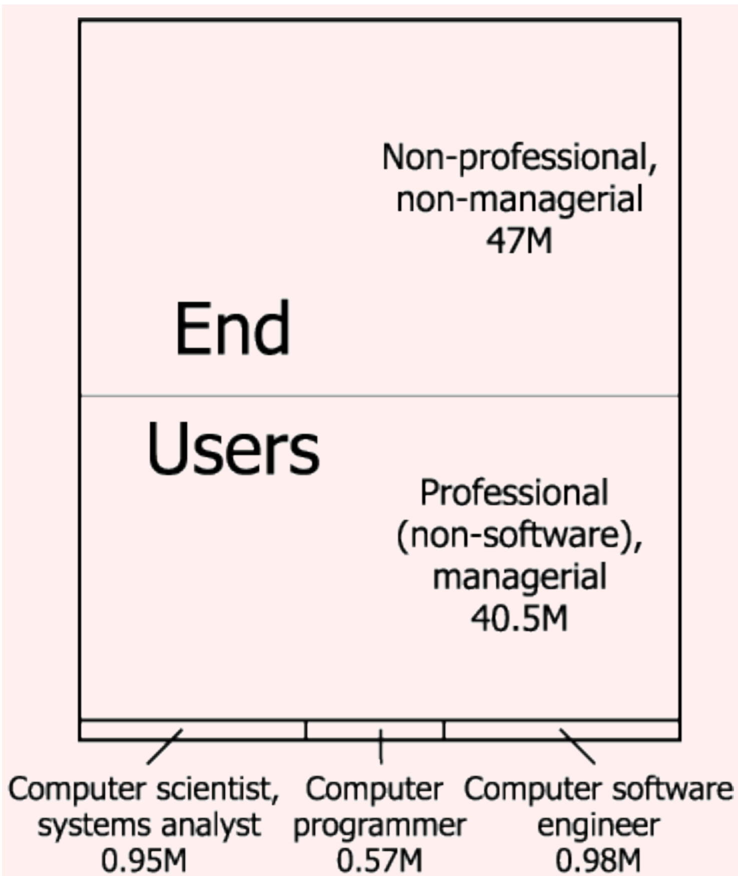
Civil order, good manners, rule of law

Empowerment of citizens to manage their own affairs

Clarity on personal security/responsibility

This requires widespread understanding of the technology and shared expectations about its use

There are *lots* of casual developers



Estimated counts in American workplace

Education

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)

C. Scaffidi, M. Shaw, and B. Myers. Estimating the Numbers of End Users and End User Programmers. *VL/HCC'05*, pp. 207-214, 2005.

http://stackoverflow.com/research/developer-survey-2015

Demographics of US Internet users

Overall	Total adults	87%
	Women	87
	Men	86
Age	18-29	97%
	30-49	93
	50-64	88
	65+	57
Geography	urban	88%
	suburban	87
	rural	83
Education	<= high school	76%
	some college	91
	college +	97

Pew Internet & American Life Project, Jan 2014 http://www.pewinternet.org/data-trend/internet-use/latest-stats/

Generations Online 2010

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

Source: Pew Internet surveys.

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

90-100%	40-49%
80-89%	30-39%
70-79%	20-29%
60-69%	10-19%
50-59%	0-9%

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

Pew Internet
Pew Internet & American Life Project

http://www.pewinternet.org/2010/12/16/generations-2010/

Email	Email	Email	Email	Email	Email
Search	Search	Search	Search	Search	Search
Health info	Health info	Health info	Health info	Health info	Health info
Social network sites	Get news	Get news	Get news	Get news	Buy a product
Watch video	Govt website	Govt website	Govt website	Travel reservations	Get news
Get news	Travel reservations	Travel reservations	Buy a product	Buy a product	Travel reservations
Buy a product	Watch video	Buy a product	Travel reservations	Govt website	Govt website
IM	Buy a product	Watch video	Bank online	Watch video	Bank online
Listen to music	Social network sites	Bank online	Watch video	Financial info	Financial info
Travel reservations	Bank online	Social network sites	Social network sites	Bank online	Religious info
Online classifieds	Online classifieds	Online classifieds	Online classifieds	Rate things	Watch video
Bank online	Listen to music	Listen to music	Financial info	Social network sites	Play games
Govt website	IM	Financial info	Rate things	Online classifieds	Online classified
Play games	Play games	IM	Listen to music	IM	Social network sites
Read blogs	Financial info	Religious info	Religious info	Religious info	Rate things
Financial info	Religious info	Rate things	IM	Play games	Read blogs
					Donate to

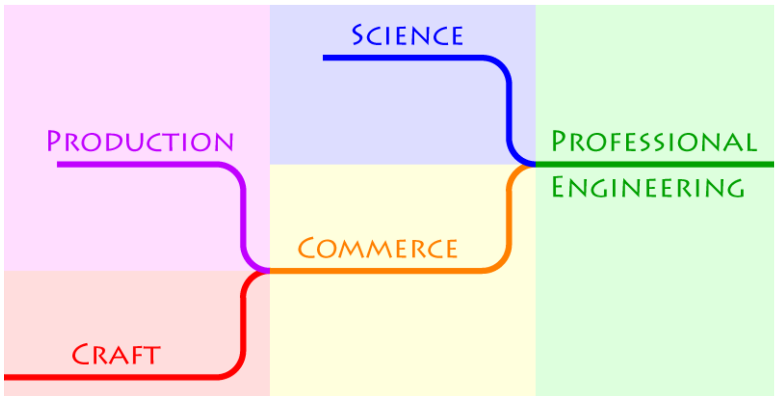
Civilizing the electronic frontier

- Policy, requiring technology
  - Balance anonymity and accountability
  - Balance security and privacy
  - Balance individual and corporate objectives
  - Address product liability
- Technology
  - Apply known best practices and designs
  - Address new forms of information access (search) and software creation (independent parts)
- User models
  - Improve the explanations and intuitions we provide the public at large

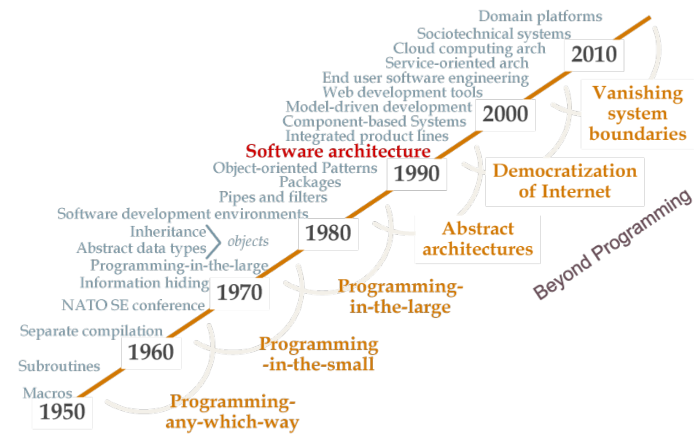

DavidGarlan
ChrisScaffidi
BradMyers
WmAWulf
RoyWeil
VahePoladian
OrnaRaz
JamesHerbsleb
ShawnButler
PaulHilfinger
RalphLondon
PhilipKoopman
RobertDeLine
PaulLi
JFTraub
JonathanRosenberg
DanielKlein
MargaretBurnett
HolgerGiese
HausiMüller
AnitaJones
WilliamScherlis
RebeccaBuchheit
PaulClements
ChristosFaloutsos
JoãoSousa
AllenCypher
MarinLitoiu
PhilipMiller
YuriyBrun
RogériodeLemos
GregoryZelesnik
MauroPezzè
JamesTomayko
PSanthanam
MichaelMauldin
AntoniaLopes
BasilBecker
AshishArora
JonLouisBentley
ArtPyster
VincenzoGrassi
MarkArdis
ThomasLane
NellyBencomo
BradleySchmerl
LawrenceFlon
JeannetteWing
DavidNotkin
RandyPausch
BonnieRay
KurtGeihs
HarryQBowik
YvonneMerrill
JWMcCredie
GiovannaDiMarzoSerugendo
DavidClark
SchahramDustdar
JeffMagee
RaffaelaMirandola
HolgerKienle
RobertFitzgerald
GuyAlmes
JamesDriscoll
JeffKramer
DanielJackson
StephenBrookes
JesperAndersson
AndrewKo
IzumiKimura
MarcDonner
SomeshJha
CristinaGacek
RonDesmarais
DannyWeyns
GaborKarsai
RajReddy
BojanCukic
CurtisScott
AlfredSpector
GaryFeldman
PaolaInverardi
ChrisBogart
SamMalek
JaneJodeit

Recapitulation

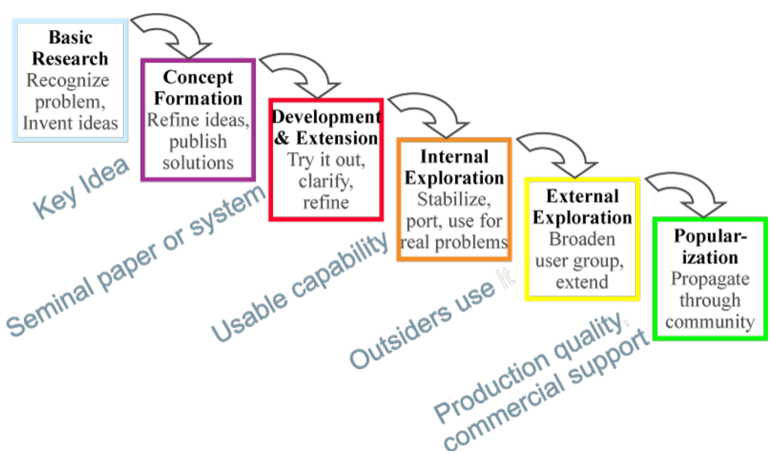
Engineering evolves from craft and commercial practice via science



Engineering basis evolves via increasingly powerful abstractions



Ideas evolve over time from pure research to practical production



The greatest need for engineering Is in the most critical applications