Computer Science & Software Engineering: Basic Science vs. Engineering?

Dieter Rombach
(dieter.rombach@iese.fraunhofer.de)

University of Kaiserslautern
Computer Science Department
Software Engineering Chair
Kaiserslautern, Germany
wwwagse.informatik.uni-kl.de

Fraunhofer Institute
for Experimental Software Engineering (IESE)
Kaiserslautern, Germany
www.iese.fhg.de
Contents

• Motivation
• Traditional Engineering
• Software Engineering
  – Research
  – Education
  – Practice
• A Scientific Foundation for SW Engineering
• World-wide Dissemination
• Future Challenges
• Outlook
Contents

- Motivation
- Traditional Engineering
- Software Engineering
  - Research
  - Education
  - Practice
- A Scientific Foundation for SW Engineering
- World-wide Dissemination
- Future Challenges
- Outlook
Motivation

• Observations
  – Importance of software is ever increasing
  – Problems with software are also increasing
Motivation (Software is crucial for business success)

Dr. S. Dais
Robert Bosch
More than 80% of all innovations in Automotive industry depend on ICT/Software.

Prof. Dr. E. Reinhardt
Siemens
In Medical Systems ICT/SW is responsible for more than 80% of all innovations.

Dr. J. Helbig
Deutsche Post
Innovations in Logistics are driven by ICT/Software to more than 80%.
Motivation (Software failures may bankrupt companies)

- Consequences from software failures increase
  - Accidents
    - Safety problems
    - Example: Child death on passenger seat of VW!
  - Loss of assets
    - Availability or security problems
    - Example: Down time of Deutsche Bank stock trading system
  - Recall actions & loss of reputation
    - Safety problems
    - Example: Recall action of car companies
Motivation

• Observations
  – Importance of software is ever increasing
  – Problems with software are also increasing
  – Software is not yet treated as an engineering artifact (most obvious & critical when combined with mechanics and electronics in cars, airplanes, etc.)
    • High quality?
    • Predictable?
    • Based on proven models?
Motivation (in-acceptable quality & predictability for critical systems)

- Most systems are too buggy
  - Standards of 2-5 defects per 1000 LoC accepted
  - This implies 10,000 – 25,000 bugs for a 5 M LoC system

- Most projects lack predictability wrt. quality/cost/time
Motivation

- Importance of software is ever increasing
  - Problems with software are also increasing
  - Software is not yet treated as an engineering artifact (most critical when combined with mechanics and electronics in cars, airplanes, etc.)
    - High quality?
    - Predictable?
    - Based on proven models?
  - Gap between research & practice is huge
  - Researchers cannot agree on big challenges (religious wars)
  - Other disciplines do “their own computer science” (not math)

- Why
  - Research focuses too much on techniques in the small (limited focus on real scientific basis & engineering)
  - Teaching treats discipline still largely as an “art”

- Deeper reasons (nature of software)
  - Software is believed to be hard to judge (no physical laws!)
  - Software is “designed” and not manufactured
  - Most software engineers have no solid education/training

---

Thesis 0: Software Engineering has some unique problems! They are not only due to our young age?
Contents

• Motivation
• Traditional Engineering
• Software Engineering
  – Research
  – Education
  – Practice
• A Scientific Foundation for SW Engineering
• World-wide Dissemination
• Future Challenges
• Outlook
Traditional (professional) Engineering

- Professional Engineering implies
  - Measurable goals
  - Best practice engineering/management principles & processes
    - DUE DILIGENCE
  - Predictability (process-product relationships, based on physical laws)
  - Reuse (regular structures, simplicity)
  - Continuous improvement
  - Explicit & validated models
Traditional Engineering

- Research areas for engineering

Thesis 1: All engineering disciplines are focusing on

- Low complexity, simplicity, regularity of products
- Predictability of processes & projects!
- Continuous improvement of the above!
Traditional Engineering

• Clear distinction between science & engineering

Mech Eng  ...  Electr. Eng.
Physics
Math

Thesis 2: All disciplines oriented towards building human-based artifacts are organized in a similar way!
Contents

• Motivation
• Traditional Engineering
• Software Engineering
  – Research
  – Education
  – Practice
• A Scientific Foundation for SW Engineering
• World-wide Dissemination
• Future Challenges
• Outlook
(Professional) Software Engineering

- Professional Software Engineering is characterized as follows
  - Often without measurable goals
  - Best practice engineering/management principles & processes are often ignored
    - DUE DILIGENCE problems
  - Predictability (process-product relationships) viewed as not possible
    - Non-determinism as human-based design discipline (up to + 150%)
  - Continuous improvement often neglected
  - Explicit & validated models largely lacking
    - “Cognitive” instead of “physical” laws are no excuse
    - Recognition of the importance of empirically based laws is only at its beginning

Thesis 3: Status of software engineering is not acceptable - especially for critical systems!
Software Engineering: State of Research

• Too much focus on tools
  – Important
  – Appropriate sequence: principles -> methods -> tools (e.g., testing)

• Too little production of evidence
  – Science is defined as “producing testable results”
  – Wrong implicit assumption of Q/C/T == f (P) leads to 150% variance
Software Engineering: State of Research

- Construction principles for complex systems (from 1970’s) exist
  - Information hiding
  - Modularization
  - Abstraction
- Lack of engineering advise (e.g., diagonalization)

---

Thesis 4: Principles for structuring complex systems are known (Parnas et al), but are rarely applied!
Software Engineering: State of Research

- Construction principles for complex systems (from 1970’s) exist
- Predictability of development processes (from 1980’s) exist

\[ T_{\text{actual}} - T_{\text{planned}} \]

Development Process

- Size of Systems
- Size of systems & Exp. of developers
- Time

Thesis 5: Principles for making design processes predictable are known (Boehm et al)
Software Engineering: State of Research

• Construction principles for complex systems (from 1970’s) exist
• Predictability of development processes (from 1980’s) exist
• Fundamentals of learning & improvement (from 1990’s) exist
  – Goal/Question/Metric Model
  – Quality Improvement Model
  – Experience Factory Model
  – Empirical methods

Thesis 6: Principles for continuous learning and improving are known (Basili et al)!
Software Engineering: State of Education

- Universities
  - Construction before analysis
    - Programming language
    - Construct programs (???)
  - Green field construction before maintenance
  - Technologies over principles
  - No emphasis on existing body of knowledge
    - No teaching of empirical studies
Handbook capturing existing body of knowledge

Students can learn about existing body of knowledge

Researcher can learn about open issues

Practitioners can avoid negligence of due diligence
Software Engineering: State of Practice (Recap)

- Most systems are too buggy
  - Standards of 2-5 defects per 1000 LoC accepted
  - This implies 10,000 – 25,000 bugs for a 5 M LoC system

- Most projects lack predictability wrt. quality/cost/time

\[
T = f_0(P) \\
T = f_1(P, C_1) \\
T = f_2(P, C_2)
\]
Software Engineering: State of Practice

• High degree of non-compliance with software (design) principles
  – Encapsulation / information hiding (e.g., Y2K)
  – Complete specification of components (e.g., 4000 non-documented interactions in a 5 M LoC system)
  – Traceability (e.g., documentation inconsistent with code)
  – ….

• High degree of non-compliance with process principles
  – Early defect detection (e.g., no sound inspections / reviews)
  – Well-understood processes ($T = f(P)$ with max 150% variance)
    – Process standards
Contents

• Motivation
• Traditional Engineering
• Software Engineering
  – Research
  – Education
  – Practice
• A Scientific Foundation for SW Engineering
• World-wide Dissemination
• Future Challenges
• Outlook
“Science” of Software Engineering

• Traditional engineering
  – Physical laws
  – Precise (prediction) models based on physical laws
  – Experimental (deterministic) evaluation

• Software engineering (recognizing the nature of software)
  – Cognitive laws
  – Empirical (prediction) models based on cognitive laws
  – Empirical (non-deterministic) evaluation
Experimental Software Engineering

- Research areas for software engineering

Experimental software engineering is pre-requisite for developing a body of knowledge relevant to the software domain!
Empirical Software Engineering

- Empirical studies are based on hypotheses testing.
- They produce results of the form "Q/P/T == f (P, C)."
- Empirical studies are to the process engineer what prototyping is to the product engineer – risk mitigation.

- They include:
  - controlled experiments in lab settings
  - case studies in field settings

- Methodologies exist, including:
  - GQM measurement approach (Basili, …)
  - QIP empirical approach (Basili, …)
  - EF experience management approach (Basili, …)

Bosch just opened the first "empirical test lab" for all their current & future practices (with IESE).
Software Engineering Structure

• Distinction between science & engineering

Thesis 7: Several arguments for separating traditional computer science into
- (Computer) Science
- (Software) Engineering
Contents

• Motivation
• Traditional Engineering
• Software Engineering
  – Research
  – Education
  – Practice
• A Scientific Foundation for SW Engineering
• World-wide Dissemination
• Future Challenges
• Outlook
Dissemination

- Most empirical software engineering research goes back to UMD (Basili et al)
- NASA’s SEL was the first “Experience Factory” for Software Engineering (Basili, McGarry, Page)
- Many Visitors came over time (some needed longer to get it!)
- This work spun off activities in many other sites
  - Fraunhofer IESE @ Germany
  - Empirical Research Group, Simula Research Lab @ Norway
  - Empirical Research Group, NICTA @ Australia
  - Et al

Thesis 8: UMD has been the mother of Experimental SE!
Vic Basili’s work created an entire discipline!

Ref: Bob Glass (CACM, 08/07)
Background – German Research Landscape

- Basic Research
  - Universities
  - 65 Max-Planck Institutes

- Applied Research & Technology Transfer
  - 58 Fraunhofer Institutes

- Industry
### Background – Fraunhofer Organization

| Named after:          | Joseph von Fraunhofer (1787-1826)  
|                       | a successful researcher, inventor and entrepreneur |
| Role of the Fraunhofer Gesellschaft: | Germany’s leading organization for applied research and technology transfer |
| Size:                 | 58 institutes with approx. 12,500 employees |
| Funding Volume:       | about 1.3 billion €, consisting of: |
| (as of 2006)          | • 1/3 base funding (government) |
|                      | • 1/3 public sector projects |
|                      | • 1/3 industrial projects |
Background – Fraunhofer Organization: ICT Group

58 Fraunhofer Institutes organized into 6 Groups

- Materials & Components
- Production Technology
- Information and Communication Technology (ICT) – D. Rombach
- Microelectronics & systems
- Energy
- Life Sciences
Software Engineering @ Kaiserslautern – IESE Highlights

- Applied Research in Software Engineering
  - Innovative methods & tools
  - Empirical evidence
- 200+ scientists & engineers
- Strategic Partner with Industry (2006: 82% external project funds)
- International Positioning
  - USA
  - Japan, India, Korea, China
  - Hungary, Ireland, Finland, …
- Innovative Cooperation Model
  - “Research Labs”
- High International Reputation in “Systems & Software Engineering”
  - No. 1 in Europe (JSS, 2005)
Software Engineering @ Kaiserslautern – IESE Competences

Quality

Cost/Time

Sustainability

Engineering Processes (requirements, architecture & component eng)

Management Processes (project & quality management)

Knowledge Management Processes & Human Resource Development (experience mgt & role-based training)
Software Engineering @ Kaiserslautern – IESE Business Areas

- Automotive, aerospace, train and shipping industry (e.g., Daimler)
- Telecom industry (e.g., T-Com)
- Medical device & health industry (e.g., Siemens)
- Financial industry (e.g., Allianz)
- E-Government & software industry (e.g., SAP)
Software Engineering @ Kaiserslautern – Fraunhofer Intern’l

USA:
- Plymouth, Michigan
- College Park, Maryland
- Boston, Massachusetts
- Newark, Delaware

CESE: Maryland, USA
- 20 FTEs
- Leads: Prof. Cleaveland,

IESE: Kaiserslautern, Germany
- 200 FTEs
- Leads: Prof. Rombach,
  Prof. Liggesmeyer
- About 75% project funding
Contents

• Motivation
• Traditional Engineering
• Software Engineering
  – Research
  – Education
  – Practice
• A Scientific Foundation for SW Engineering
• World-wide Dissemination
• Future Challenges
• Outlook
Trends & Challenges

- Software discipline needs to be structured for the future
  - Software fundamentals (science)
    - Basics: Formal notations, principles for handling software complexity, ...
    - Programming, ...
  - Software Engineering
    - Basics: Process & management models, integration of formal & semi-formal notations/techniques, empirical foundations for evidence creation, ...
    - System engineering
- Separate (and new) focus on research & teaching is needed
- New challenges need to be addressed
  - Major trends, e.g., dynamically adaptive systems
  - Challenging applications, e.g., systems of systems

**Thesis 9:** Research & Education need to be adjusted!
Research Challenges

• Listen to practical needs
• Address the entire innovation chain
  – Basic research (science), applied research (engineering) & tech transfer
  – Empirical studies are valuable vehicles at all stages
• Focus on scaleable development methods & tools
• Provide testable evaluations via empirical studies
  – For existing methods & tools
  – For newly developed methods & tools
Education Challenges

• Introduce curricula based on
  – Analyzing before constructing
    • Understanding of need for principles
    • Using “simple” construction patterns
  – Changing of systems before constructing
    • Understanding of need for principles
    • Using “simple” construction patterns
  – Focusing on principles, validating via specific technology
  – Starting with existing body of knowledge
  – Teaching of empirical studies

• Introduce separate curricula for
  – Generic software development
  – Application-specific software development
Industrial Challenges

• Adherence to “proven best practices”
  – Apply key principles
  – Choose best practices based on experience (!New Test Lab by Bosch & IESE!)
  – Educate & train personnel (e.g., IESE & University of Kaiserslautern offer distance course “Software Engineering for Engineers)

• Focus on bottleneck areas such as
  – Requirements management
  – Traceable documentation
  – Stable architectures (e.g., SOA, product lines)
  – Quality modeling & Verification / Validation
  – ...

• Employ experimental software engineering
Contents

• Motivation
• Traditional Engineering
• Software Engineering
  – Research
  – Education
  – Practice
• A Scientific Foundation for SW Engineering
• World-wide Dissemination
• Future Challenges
• Outlook
Outlook

• “Experimental” software engineering is the scientific & engineering basis professional software engineering

• Empirical studies are the pre-requisites for establishing “software laws” and understanding the potential benefits & risks of methods & tools

• Even more important as we move towards more risky applications
  – Adaptive software systems (Ambient Intelligence)
  – Globally interconnected systems of systems
  – Autonomous control systems

• Need to revise agendas for practice, research & education
Outlook

- A separation into computer science & software engineering might be the best approach to create synergies

---

**Thesis 10:** Software Engineering should develop into an area of its own (at par with other engineering disciplines)
• Dear Vic & Marv
  – We wish you a happy process of retirement!
  – We wish us that this process never ends!