

Experimental Software Engineering: Building the Scientific Basis for Professional Software Engineering

Dieter Rombach

(dieter.rombach@iese.fraunhofer.de)

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

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

Contents

- **Software Engineering @ Kaiserslautern**
- **Introduction & Motivation**
- **Current Status**
 - Practice
 - Research
 - Education
- **Professional Software Engineering**
- **“Science” of Software Engineering**
- **Experimental Software Engineering**
- **Empirical Software Engineering**
- **Challenges**
 - Industry
 - Research
 - Education
- **Outlook**

Contents

- **Software Engineering @ Kaiserslautern**
- **Introduction & Motivation**
- **Current Status**
 - Practice
 - Research
 - Education
- Professional Software Engineering
- “Science” of Software Engineering
- Experimental Software Engineering
- Empirical Software Engineering
- Challenges
 - Industry
 - Research
 - Education
- Outlook

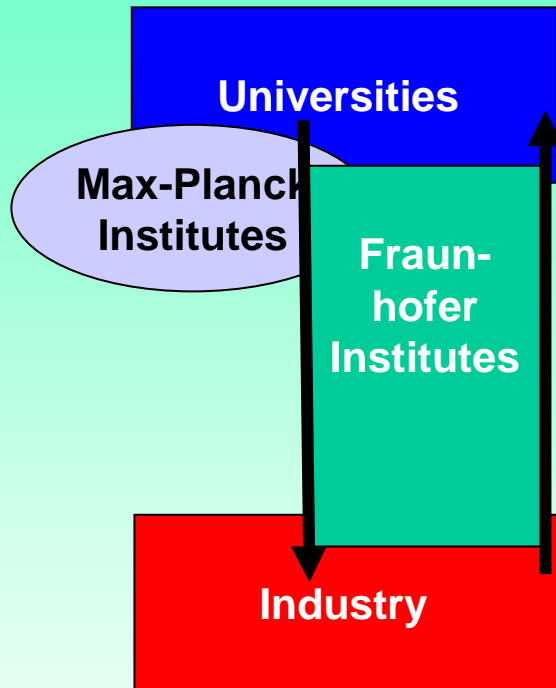
Software Engineering @ Kaiserslautern - Location

Central Location in Europe:

- South-West of Germany
- 1 h from Frankfurt
- close to France, Belgium, and Luxembourg



Background – German Research Landscape



- Basic Research
 - Universities
 - 65 Max-Planck Institutes
- Applied Research & Technology Transfer
 - 58 Fraunhofer Institutes
- Industry

German Research Landscape focuses on complete innovation chain!

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: (as of 2006)	about 1.3 billion €, consisting of: <ul style="list-style-type: none">• 1/3 base funding (government)• 1/3 public sector projects• 1/3 industrial projects

Fraunhofer Gesellschaft is Europe's largest Applied Research Organization!

Background – Fraunhofer Organization: Naming



Researcher

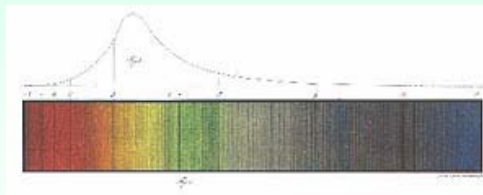
discovery of “Fraunhofer Lines” in the sun’s spectrum

Inventor

new methods of lens processing

Entrepreneur

head of royal glass factory



Joseph von Fraunhofer (1787-1826)

- Name patron as 1st German Scientist who made money from science
- Exhibition in „Zwinger“ Museum in Leipzig, Germany

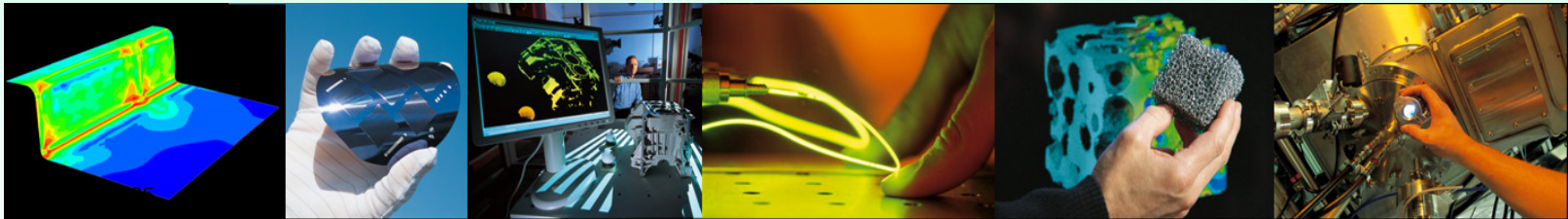
Background – Fraunhofer Organization: ICT Group

Fraunhofer ICT Group

- 15 institutes
- 3000 scientists

58 Fraunhofer Institutes organized into 6 Groups

- Materials & Components
- Production Technology
- Information and Communication Technology (ICT) – D. Rombach
- Microelectronics & systems
- Energy
- Life Sciences



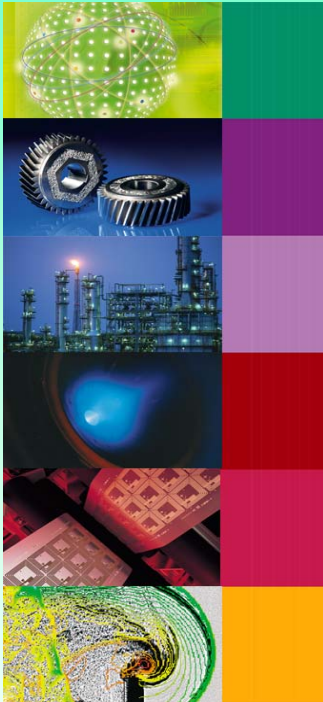
Software Engineering @ Kaiserslautern – ICT Alliance

- Complete innovation chain in ICT / Software @ Kaiserslautern
- Highly rated basic research
 - University with highly ranked (top-5) departments in **computer science, electric engineering & mathematics**
 - **Max-Planck Institute** for Software Systems
- Internationally ranked applied research
 - 2 **Fraunhofer Institutes** for software engineering (IESE) and simulation & optimization (ITWM)
 - German Center for Artificial Intelligence (DFKI)
- Innovative transition & binding
 - **Research Labs** with International companies
 - ICT & service companies (**4000 new jobs in 5 years**)

Kaiserslautern's ICT Alliance:

- 800 scientists
- coverage of innovation chain

Software Engineering @ Kaiserslautern – IESE Mission

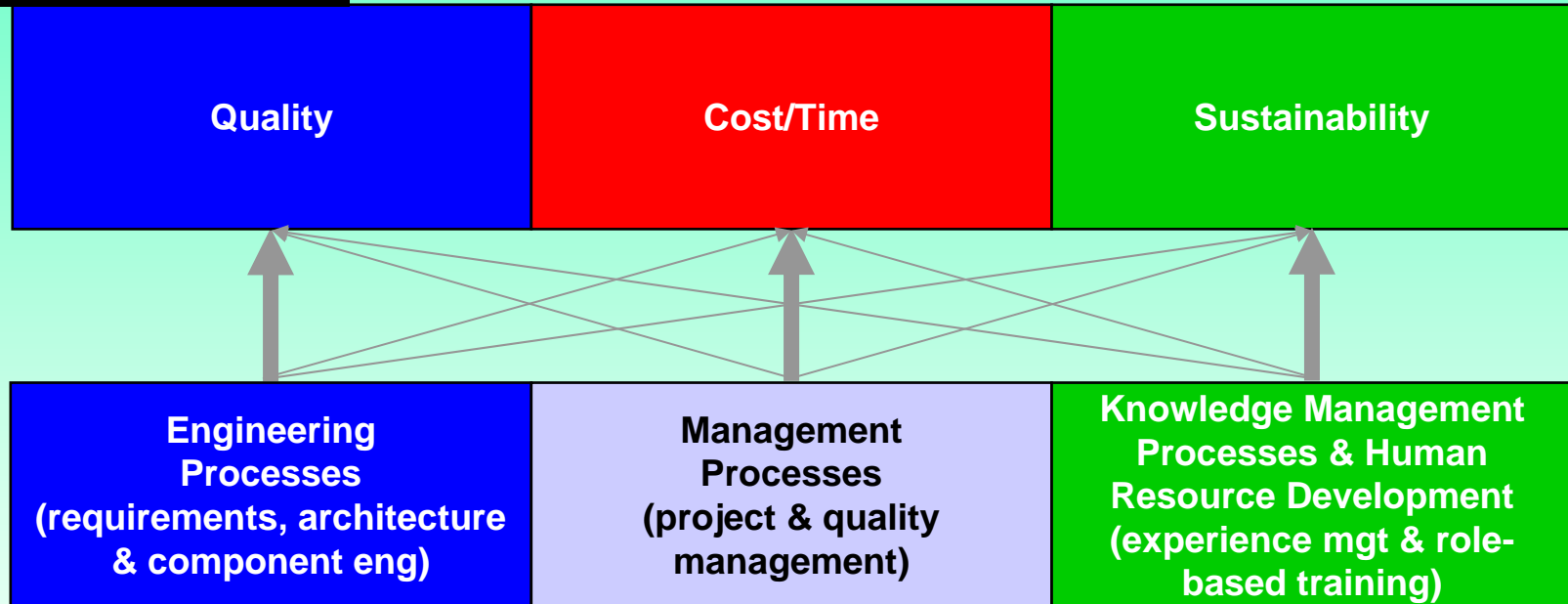


- Provide innovative and value-adding customer solutions **with measurable effects**
- Advance the state-of-the art in software & systems engineering
- Promote the importance of **empirically based software & systems engineering**

Experimental Software Engineering

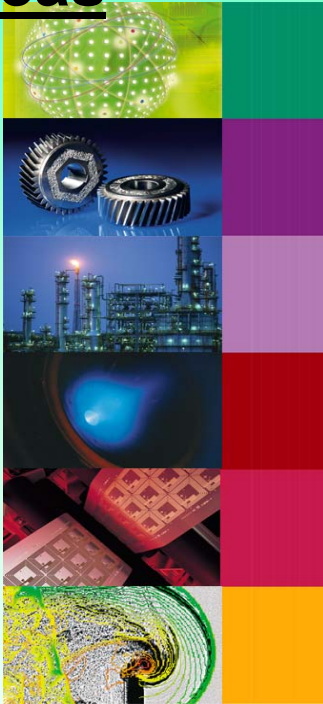
Software Engineering @ Kaiserslautern – IESE

Competences



All 3 competences are needed for sustained industrial success!

Software Engineering @ Kaiserslautern – IESE Business Areas

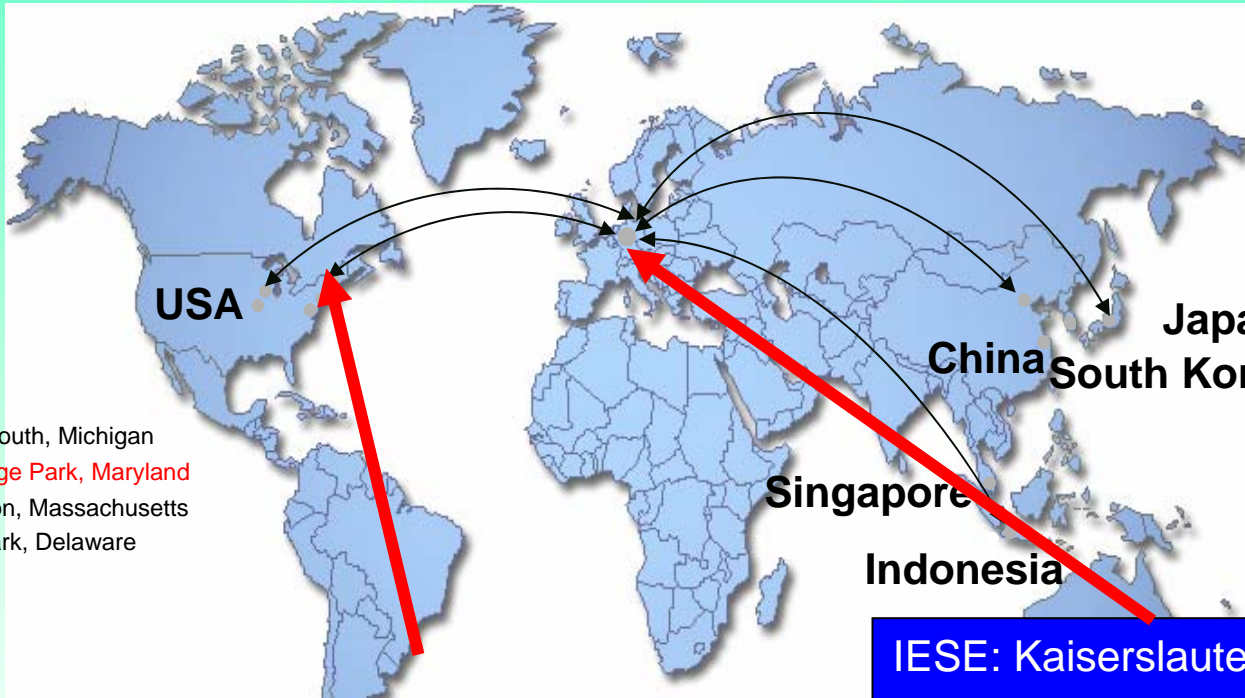


- Automotive, aerospace, train and shipping industry (e.g., Daimler)
- Telecom industry (e.g., T-Com)
- Medical device & health industry (e.g., Siemens)
- Financial industry (e.g., Allianz)
- E-Government & software industry (e.g., SAP)

Focus is on industries based on embedded software or information systems with high quality of service requirements!



Software Engineering @ Kaiserslautern – Fraunhofer Intern'l



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

Fraunhofer institutes (e.g., IESE) have high international presence!

CESE: Maryland, USA

- 25 FTEs
- Leads: Prof. Cleaveland, Prof. Basili

IESE: Kaiserslautern, Germany

- 200 FTEs
- Leads: Prof. Rombach, Prof. Liggesmeyer
- About 75% project funding

Software Engineering @ Kaiserslautern – IESE Highlights

- 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)

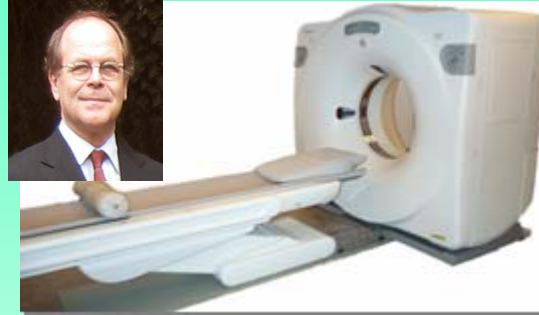


IESE is one of the most successful Fraunhofer institutes (e.g., publications, income, spinoffs)!

Introduction & Motivation



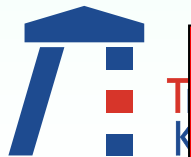
Dr. S. Dais
Robert Bosch
More than **80%** of all innovations in **Automotive industry** depend on **ICT/Software**.



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%**.



Software is crucial for business success!

Introduction & Motivation

- **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

Software failures may bankrupt companies!

Introduction & Motivation

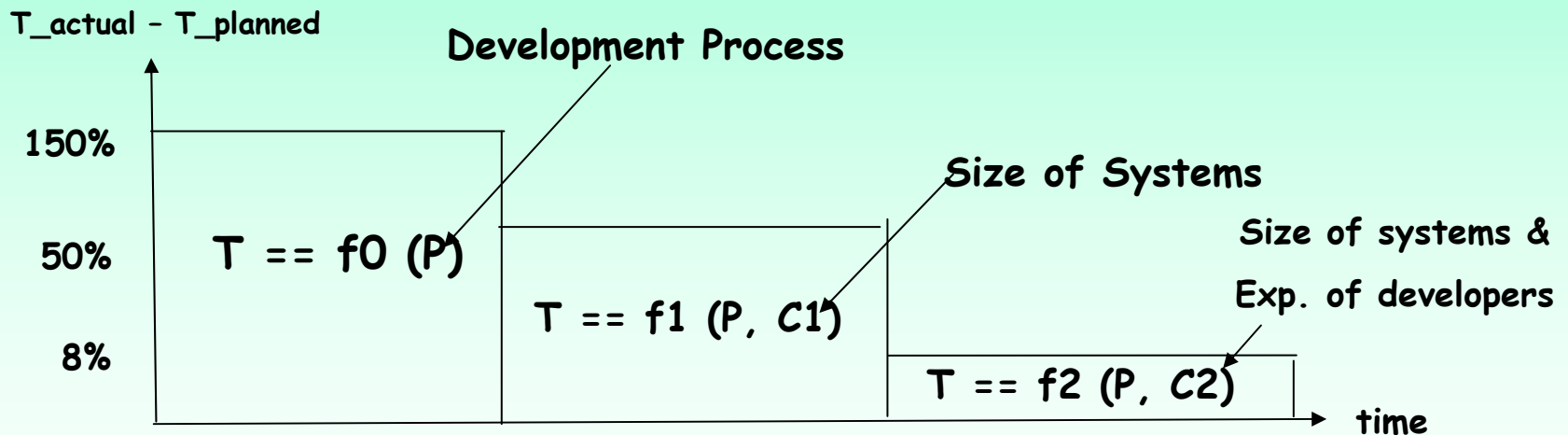
- **Current software development practices are sub-optimal**
 - **Wide-spread perception**
 - **Art instead of engineering**
 - Results hard to predict
 - Software is “buggy” by nature
 - Supported by many assessments
- **Satisfactory state-of-the-art practices exist**
 - Requires change of mind
 - Current practice not acceptable
 - “proven” best practices exist
 - Supported by some leading companies

Contents

- **Software Engineering @ Kaiserslautern**
- **Introduction & Motivation**
- **Current Status**
 - Practice
 - Research
 - Education
- Professional Software Engineering
- “Science” of Software Engineering
- Experimental Software Engineering
- Empirical Software Engineering
- Challenges
 - Industry
 - Research
 - Education
- Outlook

State of Practice

- **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**



**Inacceptable Status for
Critical Systems!**

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**)

**Lack of Due Diligence!
Lack of Professionalism!**

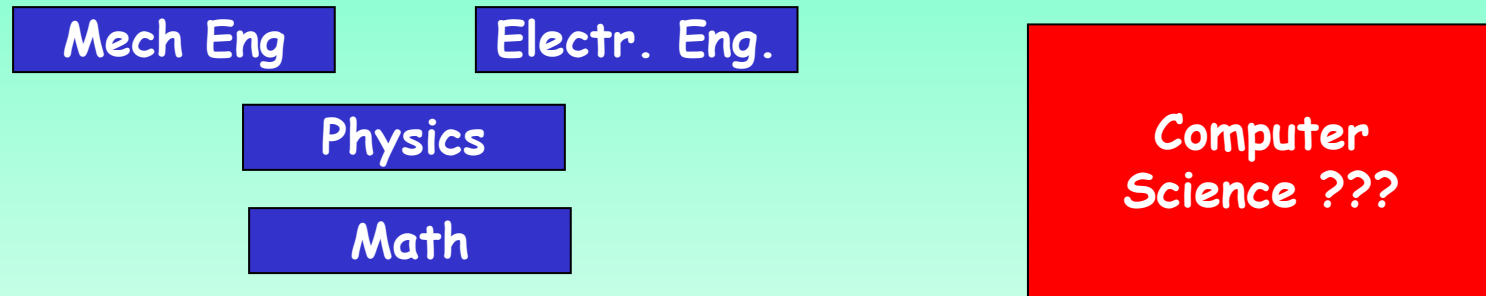
State of Research

- **Too much focus on tools**
 - Important
 - Sensible sequence: principles -> methods -> tools
- **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

**Inacceptable production of technologies
without any evidence of usage risk!**

State of Research

- Not clearly differentiated between science & engineering



- Methods often lack
 - scalability wrt. complexity
 - Scalability wrt. notation
- Gap between research & practice is too large, and is in danger of growing further

Mixing of science & engineering
in the software domain
creates religious wars instead
of producing solutions!



State of Education

- **Schools**
 - Too much “engineering”
 - Too little underlying science (e.g, discrete math, alg)
- **Public perception about “job profile” for computer science**
 - Programming
 - No separation between skill-based activities (e.g., programming) & engineering-based activities (e.g., architect, quality assurance)

**Wrong professional perception
created in public!**

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

Re-enforcement of „Art“ perception!

Contents

- **Software Engineering @ Kaiserslautern**
- **Introduction & Motivation**
- **Current Status**
 - Practice
 - Research
 - Education
- **Professional Software Engineering**
- **“Science” of Software Engineering**
- **Experimental Software Engineering**
- **Empirical Software Engineering**
- **Challenges**
 - Industry
 - Research
 - Education
- **Outlook**

Professional Software Engineering

- Professional Engineering implies
 - Measurable goals
 - Best practice engineering/management principles & processes
 - DUE DILIGENCE
 - Predictability (process-product relationships)
 - Continuous improvement
 - Explicit models (before implementation)
- Professional **Software** engineering implies
 - Best practice principles
 - Predictability (process-product relationships)
 - Non-determinism as human-based design discipline (+ X%)
 - Explicit models
 - Of empirical nature
 - Software is based on “cognitive” instead of “physical” laws



**Empirical Nature of Software Engineering is
no excuse for Unpredictability!**

KAISERSLAUTERN

Experimentelles
Software Engineering

“Science” of Software Engineering

- **Traditional engineering**
 - Physical laws
 - Precise (prediction) models based on physical laws
 - Experimental (deterministic) evaluation
- **Software engineering**
 - Cognitive laws
 - Empirical (prediction) models based on cognitive laws
 - Empirical (non-deterministic) evaluation

**Engineering of software
requires an understanding of
the nature of software**

'giving the reader a proven basis for engineering complex software systems'

A Handbook of Software and Systems Engineering

Empirical Observations, Laws and Theories

Albert Endres
Dieter Rombach

Fraunhofer
Institut
Experimentelles
Software Engineering

PEARSON
Addison
Wesley

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

“Science” of Software Engineering

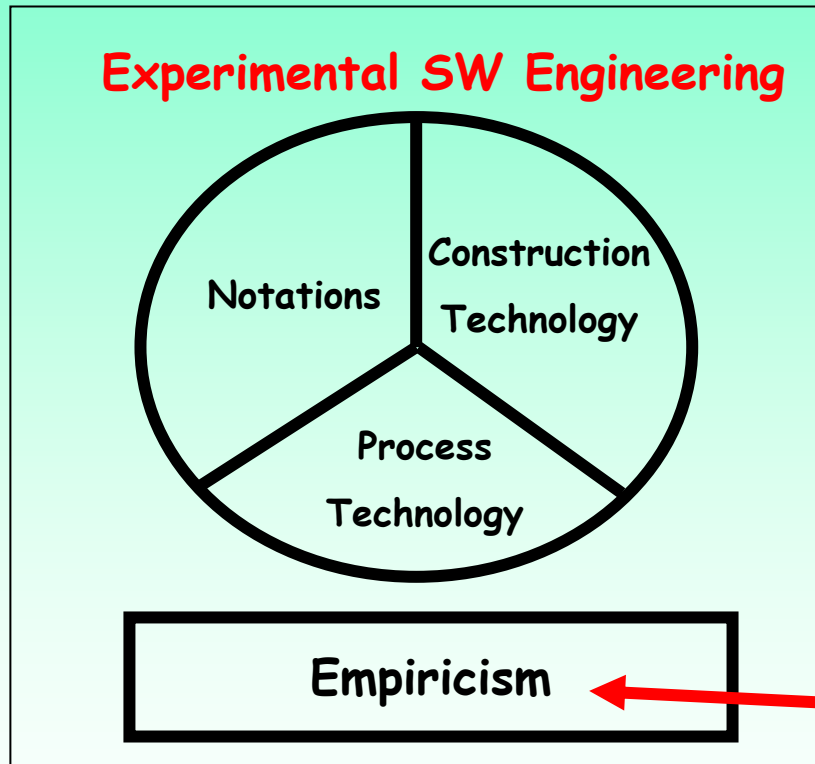
- **Engineering**
 - **Math:** continuous math
 - **Science:** physics
 - **Engineering:** mech. eng, electr. eng, ...
- **Software Engineering**
 - **Math:** discrete math
 - **Science:** computer science, economics, statistics
 - **Engineering:** software engineering

Embedded System engineering

Field of „Computer Science“ needs to be re-structured in order to reflect needs

Experimental Software Engineering

- Research paradigm for software engineering



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

Empirical Software Engineering

Empirical Software Engineering

- Empirical studies are based on hypotheses testing
- They produce results of the form “**Q/P/T == f (P, C)**”
- They include
 - controlled experiments in lab settings
 - case studies in field settings
- Method box exists, including
 - GQM measurement approach (Basili, Rombach, ...)
 - QIP empirical approach (Basili, Rombach, ...)
 - EF experience management approach (Basili, Rombach, ...)
- Best known empirical settings
 - NASA SEL, USA
 - Fraunhofer IESE & CESE, Germany & USA
 - Simula Research Lab, Norway
 - NICTA Empirical Group, Australia

Ref: Bob Glass
(CACM, 08/07)

Contents

- **Software Engineering @ Kaiserslautern**
- **Introduction & Motivation**
- **Current Status**
 - Practice
 - Research
 - Education
- **Professional Software Engineering**
- **“Science” of Software Engineering**
- **Experimental Software Engineering**
- **Empirical Software Engineering**
- **Challenges**
 - Industry
 - Research
 - Education
- **Outlook**

Industrial Challenges

- Adherence to “proven best practices”
 - Apply key principles
 - **Choose best practices based on experience**
 - 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**



**Adoption of professional engineering discipline
to software engineering is needed!**

IESE Offerings to address Industrial Challenges

- **Software product line approach (PuLSE)**
 - Increase of reuse level (up to 90%)
 - Increase in certifiable quality
 - Customers: Daimler, Bosch, Ricoh, Siemens, T-COM, ...
- **Measurement-based project & quality management**
 - Predictability (+/- 5-8% on target)
 - Early risk identification
 - Customers: see above
- **Empirical research labs**
 - Best practice demonstration
 - Education
 - Technology evaluation
 - Customers: ... Bosch builds 1st distributed empirical test lab!



**Fraunhofer IESE supports key areas
to professionalize software engineering**

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

Lets take „engineering“ of software serious!

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**

Lets teach pride in „simple“ solutions!

Outlook

- **“Experimental” software engineering is 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 information systems**
 - **Autonomous control systems**
- **Need to revise agendas for practice, research & education**



There is no alternative!

