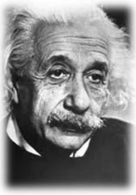
Prof. Dr. Harald Störrle
Danmarks Tekniske Universitet (DTU)

Glaube, Liebe, Hoffnung:
Requirements Engineering State of the Art
Aus Sicht der Evidenz-basierten Informatik

AK RE München, Fa. method park, 15.12.2014

Alternative Ways to Convince People

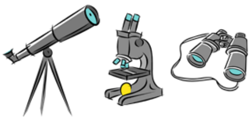
By force
physical force, group pressure

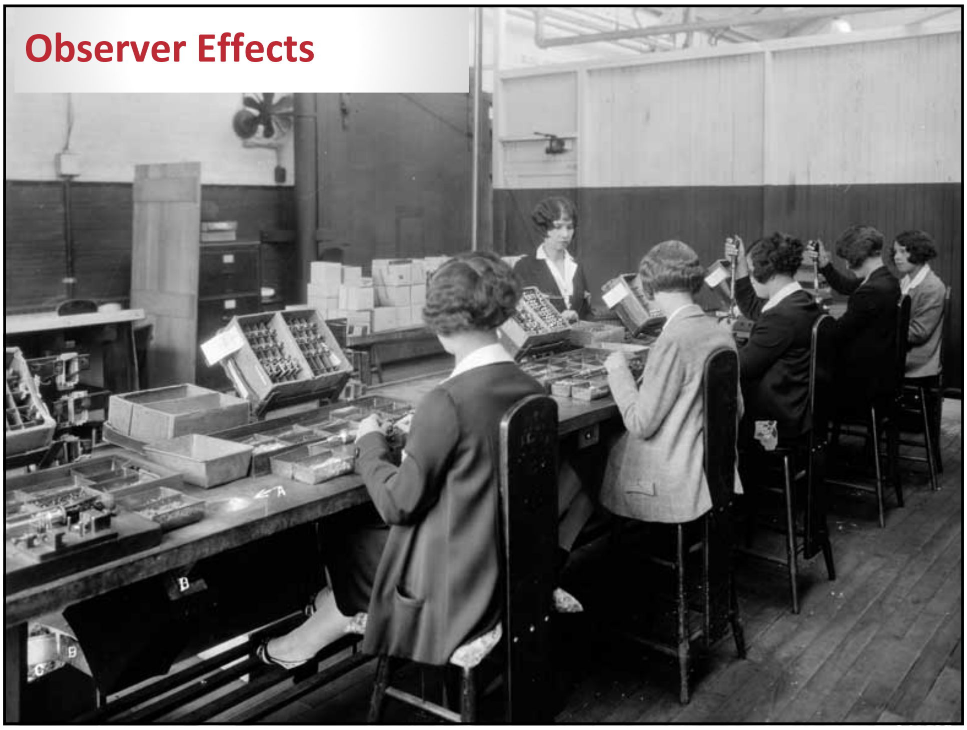


By authority
divine revelation, fame

By insight
plausibility, observation

By research





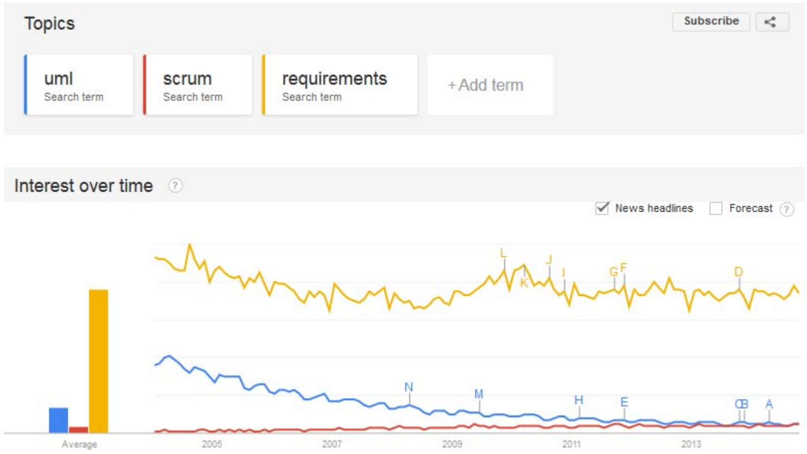
Observer Effects

Prof. Dr. Harald Störrle
Danmarks Tekniske Universitet (DTU)

Motivation & Justification

RE – a relevant topic?

- Let‘s have a quick look at Google Trends for a first impression.

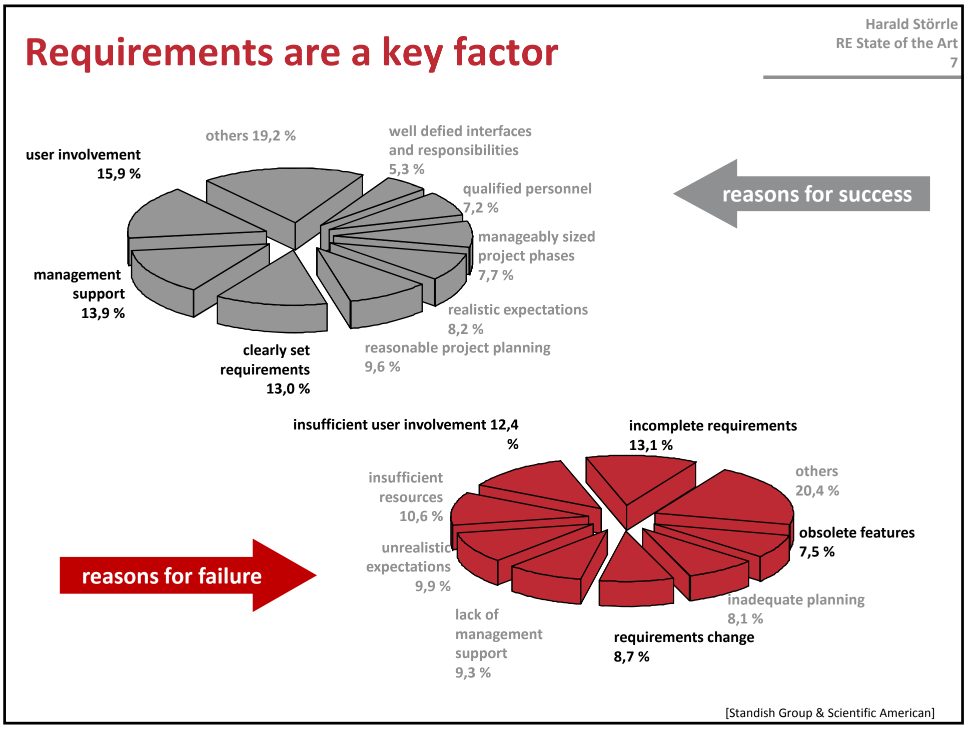


More (reliable) sources

- **Requirements Engineering (or the lack thereof) is still the single most important reason for poor software quality**
  - “Lutz [1993] showed that 60% of errors in critical systems were the results of requirements errors.
  - Espiti [1996] conducted a survey of European companies and found that more than 60% of them considered requirements engineering problems as very significant.
  - Hall et al. [2002] carried out a case study of 12 companies at different CMM levels. They discovered that, out of a total of 268 development problems cited, almost 50% (128) were requirements problems.”

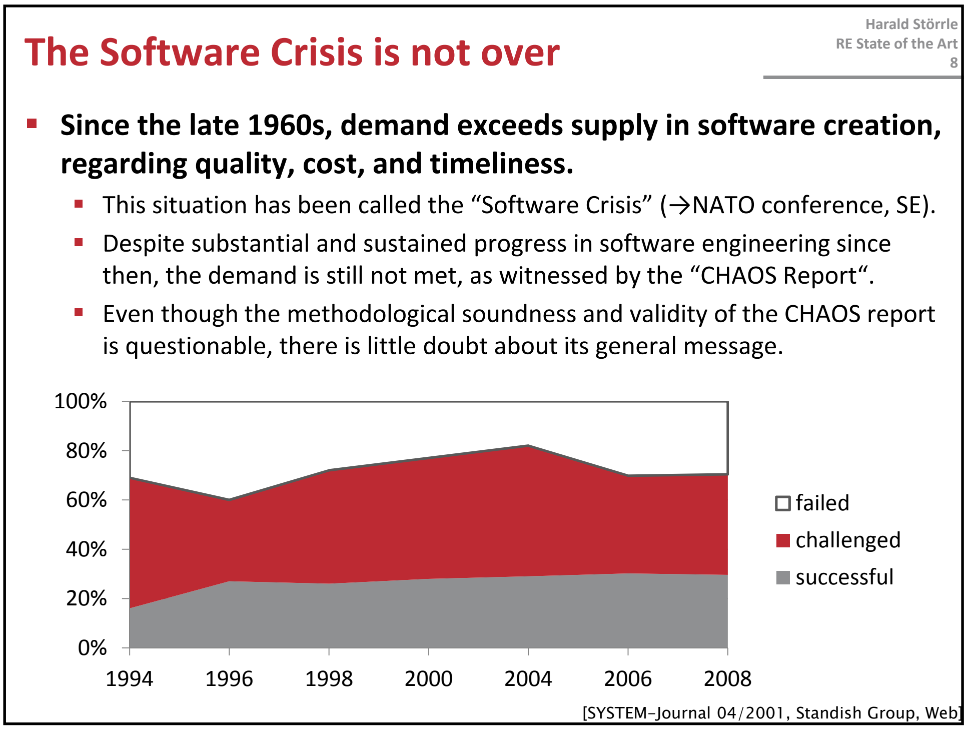
- „Nonetheless, requirements engineering is still performed in an intuitive and chaotic way.”

Sommerville, I., Ransom, J.: An Empirical Study of Industrial Requirements Engineering Process Assessment and Improvement. ACM Transactions on Software Engineering and Methodology 14(1), 85–117 (2005)



Requirements are a key factor

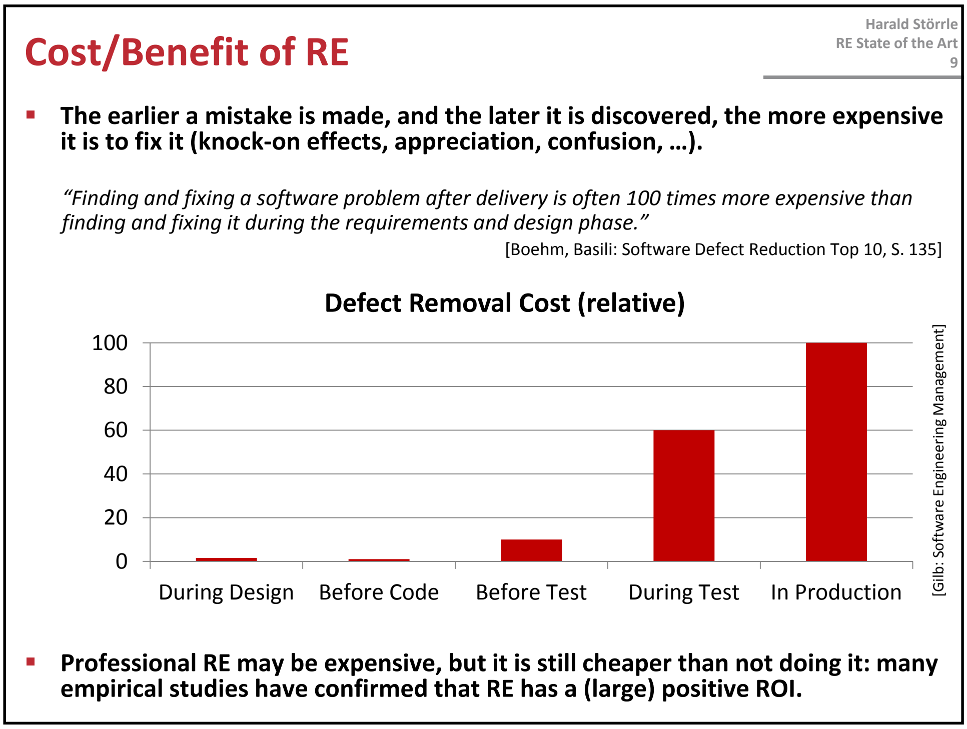
[Standish Group & Scientific American]



The Software Crisis is not over

- **Since the late 1960s, demand exceeds supply in software creation, regarding quality, cost, and timeliness.**
  - This situation has been called the “Software Crisis” (→NATO conference, SE).
  - Despite substantial and sustained progress in software engineering since then, the demand is still not met, as witnessed by the “CHAOS Report“.
  - Even though the methodological soundness and validity of the CHAOS report is questionable, there is little doubt about its general message.

[SYSTEM-Journal 04/2001, Standish Group, Web]



Cost/Benefit of RE

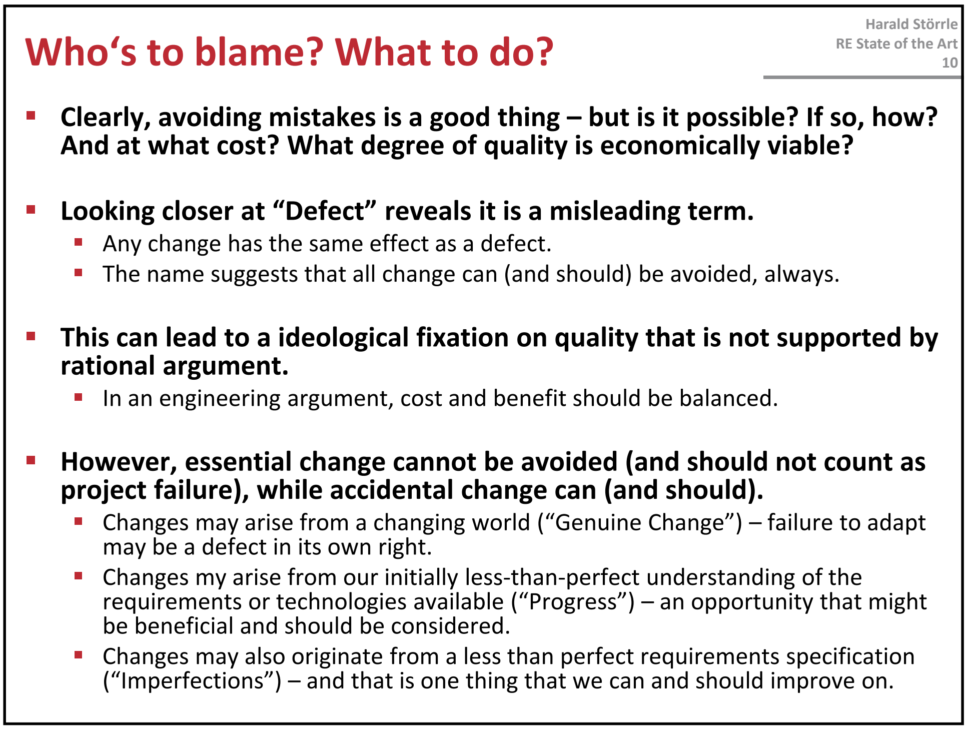
- **The earlier a mistake is made, and the later it is discovered, the more expensive it is to fix it (knock-on effects, appreciation, confusion, …).**

"Finding and fixing a software problem after delivery is often 100 times more expensive than finding and fixing it during the requirements and design phase."
[Boehm, Basili: Software Defect Reduction Top 10, S. 135]

Defect Removal Cost (relative)

[Gilb: Software Engineering Management]

- **Professional RE may be expensive, but it is still cheaper than not doing it: many empirical studies have confirmed that RE has a (large) positive ROI.**

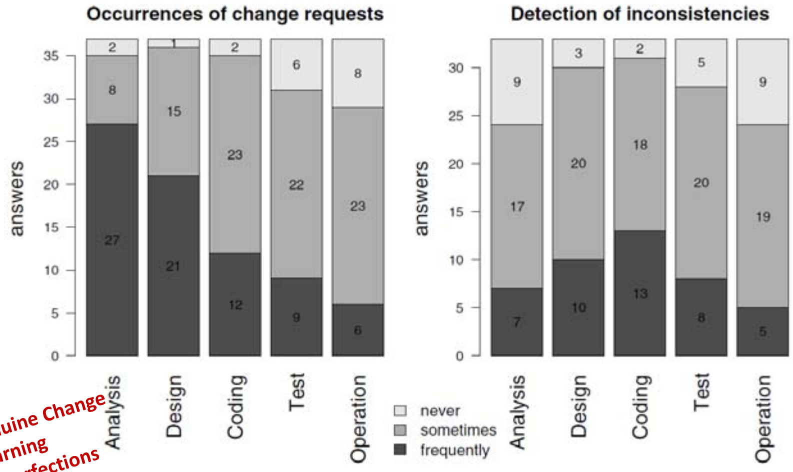


Who's to blame? What to do?

- **Clearly, avoiding mistakes is a good thing – but is it possible? If so, how? And at what cost? What degree of quality is economically viable?**
- **Looking closer at "Defect" reveals it is a misleading term.**
  - Any change has the same effect as a defect.
  - The name suggests that all change can (and should) be avoided, always.
- **This can lead to a ideological fixation on quality that is not supported by rational argument.**
  - In an engineering argument, cost and benefit should be balanced.
- **However, essential change cannot be avoided (and should not count as project failure), while accidental change can (and should).**
  - Changes may arise from a changing world ("Genuine Change") – failure to adapt may be a defect in its own right.
  - Changes my arise from our initially less-than-perfect understanding of the requirements or technologies available ("Progress") – an opportunity that might be beneficial and should be considered.
  - Changes may also originate from a less than perfect requirements specification ("Imperfections") – and that is one thing that we can and should improve on.

Looking closer at „Defects“

- Analyzing the origins of “defects” requests can help us understand their nature.



(a) Genuine Change
(b) Learning
(c) Imperfections

Stefan Winkler: **Information Flow Between Requirement Artifacts. Results of an Empirical Study**
P. Sawyer, B. Paech, and P. Heymans (Eds.): REFSQ 2007, LNCS 4542, pp. 232–246, Springer, 2007

Prof. Dr. Harald Störrle
Danmarks Tekniske Universitet (DTU)

Requirements in Industry

RE-Process Maturity Frameworks exist

- **The REAIMS framework was published in the mid 1990s.**

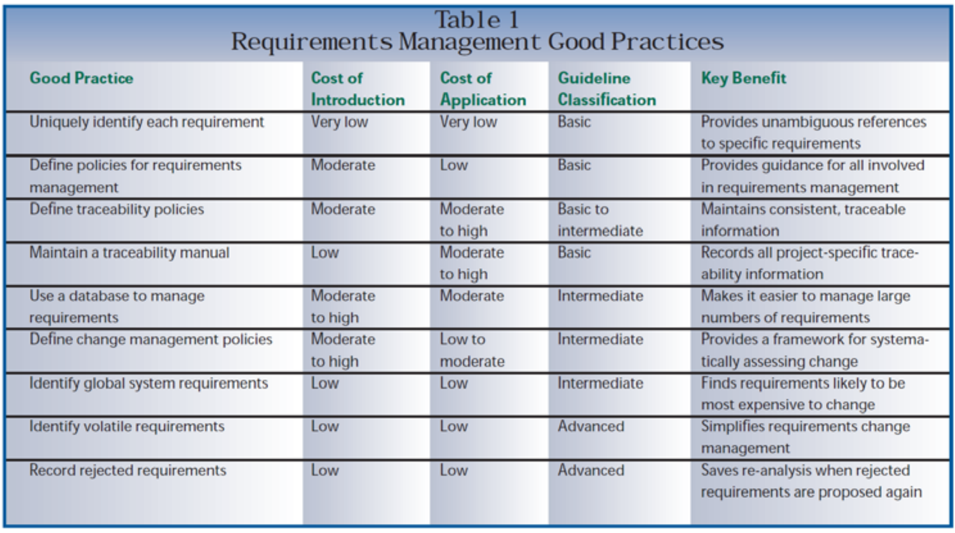


Table 1
Requirements Management Good Practices

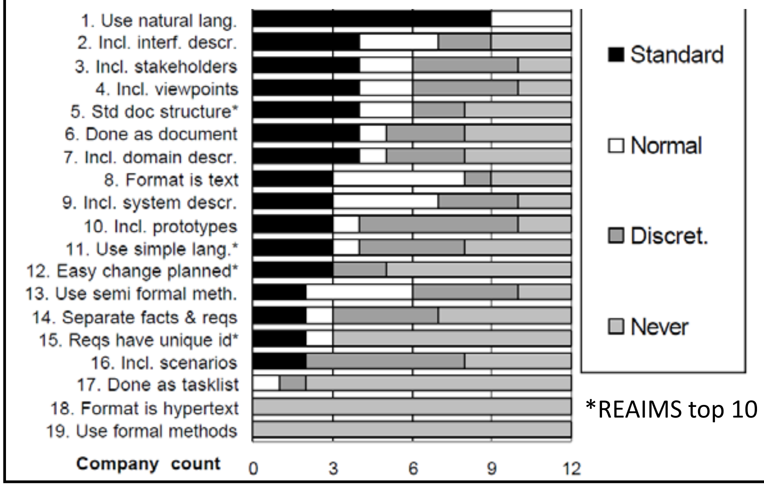
Good Practice	Cost of Introduction	Cost of Application	Guideline Classification	Key Benefit
Uniquely identify each requirement	Very low	Very low	Basic	Provides unambiguous references to specific requirements
Define policies for requirements management	Moderate	Low	Basic	Provides guidance for all involved in requirements management
Define traceability policies	Moderate	Moderate to high	Basic to intermediate	Maintains consistent, traceable information
Maintain a traceability manual	Low	Moderate to high	Basic	Records all project-specific traceability information
Use a database to manage requirements	Moderate to high	Moderate	Intermediate	Makes it easier to manage large numbers of requirements
Define change management policies	Moderate to high	Low to moderate	Intermediate	Provides a framework for systematically assessing change
Identify global system requirements	Low	Low	Intermediate	Finds requirements likely to be most expensive to change
Identify volatile requirements	Low	Low	Advanced	Simplifies requirements change management
Record rejected requirements	Low	Low	Advanced	Saves re-analysis when rejected requirements are proposed again

Sawyer, Pete, Sommerville, Ian, Viller, Stephen: Capturing the benefits of requirements engineering. IEEE Softw. 16,2, 78–85, IEEE 1997

State of RE in Practice is Poor

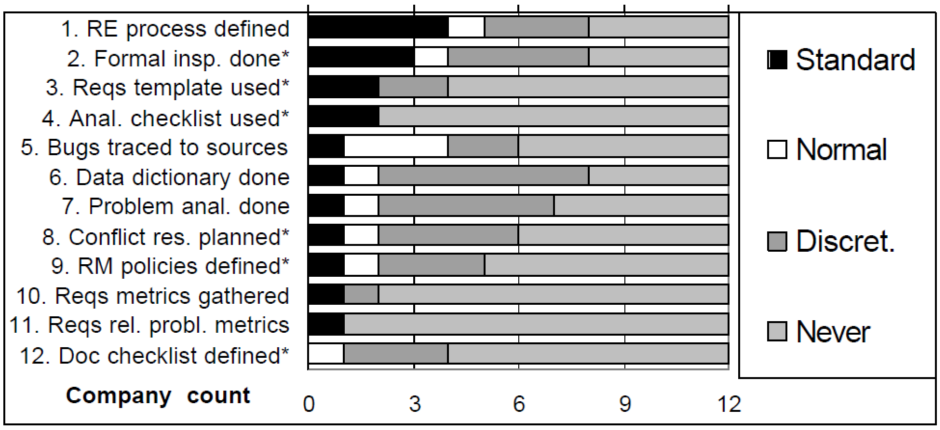
- *“There is a lot of information available on solid RE practices but anecdotal evidence still indicates poor practices.”*

U. Nikula, J. Sajaniemi, H. Kälviäinen: A State-of-the-Practice Survey on Requirements Engineering in Small- and Medium-Sized Enterprises. Telecom Business Research Center Lappeenranta, Research Report 1, 2000



*REAIMS top 10

RE Process Maturity in Practice is Poor

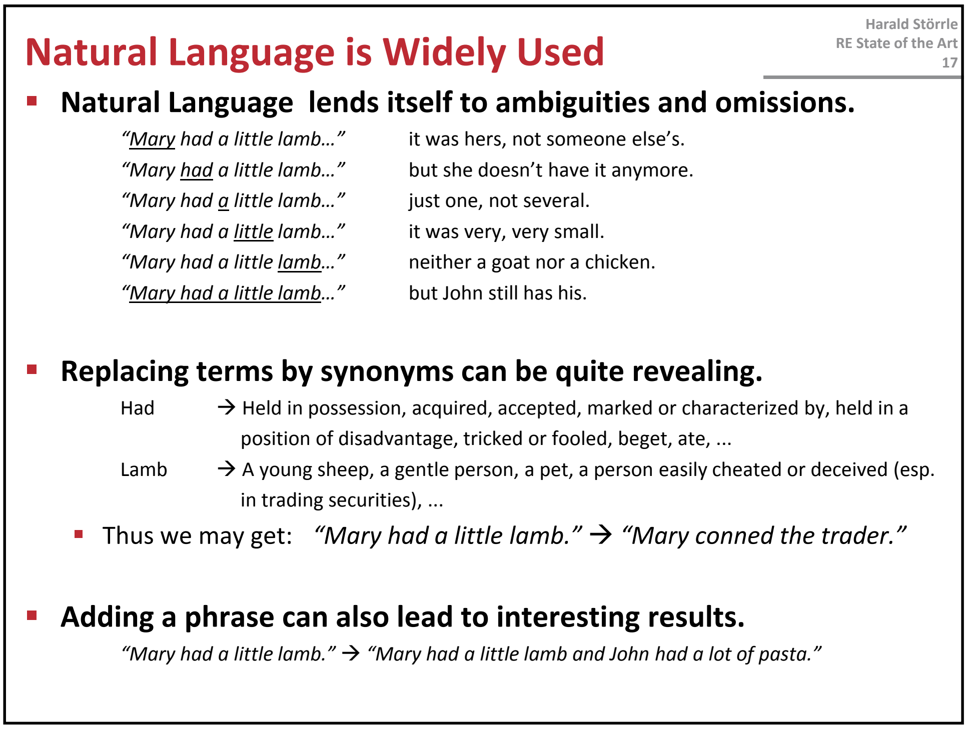


*REAIMS top 10

Improving RE Process Maturity is easy

- Here are some examples of the practices by maturity level defined by the REAIMS RE process maturity framework.
- Basic
  - 3.1 Define a standard document structure
  - 4.3 Identify and consult system stakeholders
  - 6.2 Use language simply, consistently and concisely
  - 8.2 Organize formal requirements inspections
- Intermediate
  - 4.10 Prototype poorly understood requirements
  - 9.6 Define change management policies
- Advanced
  - 10.6 Specify systems using formal specifications
  - 10.8 Collect incident experience

[Ian Sommerville, Pete Sawyer: *Requirements Engineering: A Good Practice Guide*. Wiley, 1997]



Natural Language is Widely Used

- **Natural Language lends itself to ambiguities and omissions.**

“Mary had a little lamb…”	it was hers, not someone else’s.
“Mary had a little lamb…”	but she doesn’t have it anymore.
“Mary had a little lamb…”	just one, not several.
“Mary had a little lamb…”	it was very, very small.
“Mary had a little lamb…”	neither a goat nor a chicken.
“Mary had a little lamb…”	but John still has his.

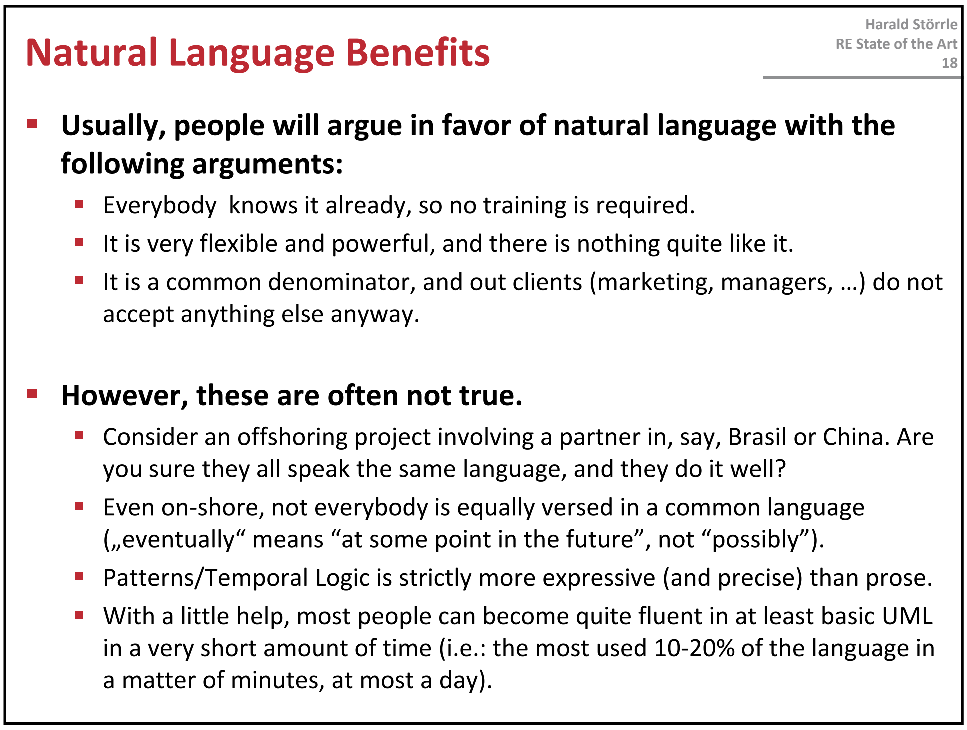
- **Replacing terms by synonyms can be quite revealing.**

Had	→ Held in possession, acquired, accepted, marked or characterized by, held in a position of disadvantage, tricked or fooled, beget, ate, ...
Lamb	→ A young sheep, a gentle person, a pet, a person easily cheated or deceived (esp. in trading securities), ...

  - Thus we may get: *“Mary had a little lamb.”* → *“Mary conned the trader.”*

- **Adding a phrase can also lead to interesting results.**

  “Mary had a little lamb.” → *“Mary had a little lamb and John had a lot of pasta.”*



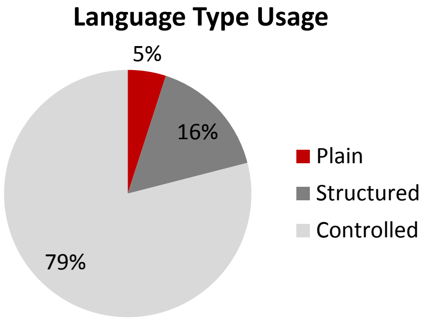
Natural Language Benefits

- **Usually, people will argue in favor of natural language with the following arguments:**
  - Everybody knows it already, so no training is required.
  - It is very flexible and powerful, and there is nothing quite like it.
  - It is a common denominator, and out clients (marketing, managers, …) do not accept anything else anyway.

- **However, these are often not true.**
  - Consider an offshoring project involving a partner in, say, Brasil or China. Are you sure they all speak the same language, and they do it well?
  - Even on-shore, not everybody is equally versed in a common language („eventually“ means “at some point in the future”, not “possibly”).
  - Patterns/Temporal Logic is strictly more expressive (and precise) than prose.
  - With a little help, most people can become quite fluent in at least basic UML in a very short amount of time (i.e.: the most used 10-20% of the language in a matter of minutes, at most a day).

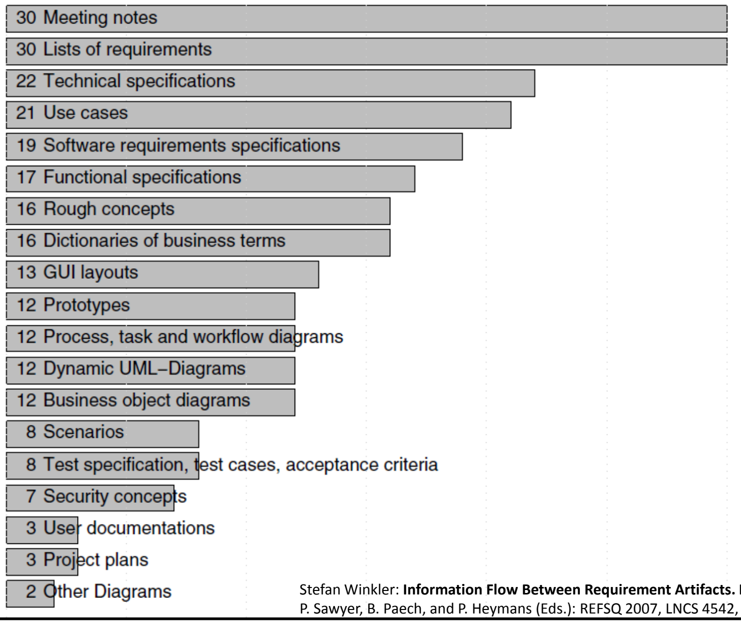
Prose for Requirements Engineering

- Alternatives exist, that can (mostly) replace NL, as various case studies have demonstrated.



- Natural Language Processing (NLP) and Information Retrieval (IR) technology can do amazing things:
  - generating sequence diagrams from natural language use case descriptions;
  - generating class diagrams from NL requirements specifications.
- However, if the performance is less than perfect, using tools is often worse than not using them.

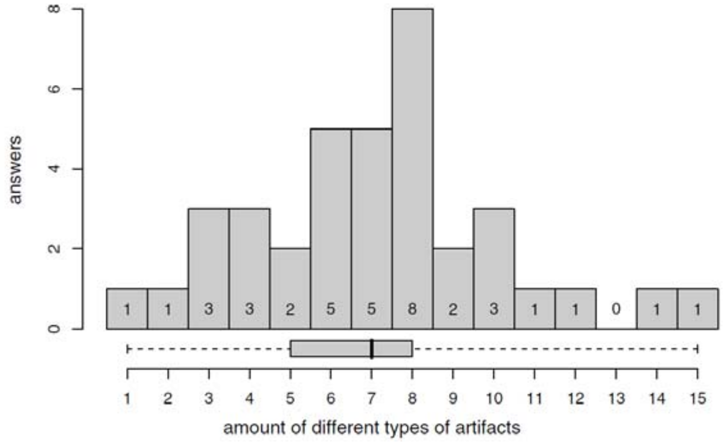
Diverse Requirements Stores in use



Stefan Winkler: **Information Flow Between Requirement Artifacts. Results of an Empirical Study**
P. Sawyer, B. Paech, and P. Heymans (Eds.): REFSQ 2007, LNCS 4542, pp. 232–246, Springer, 2007

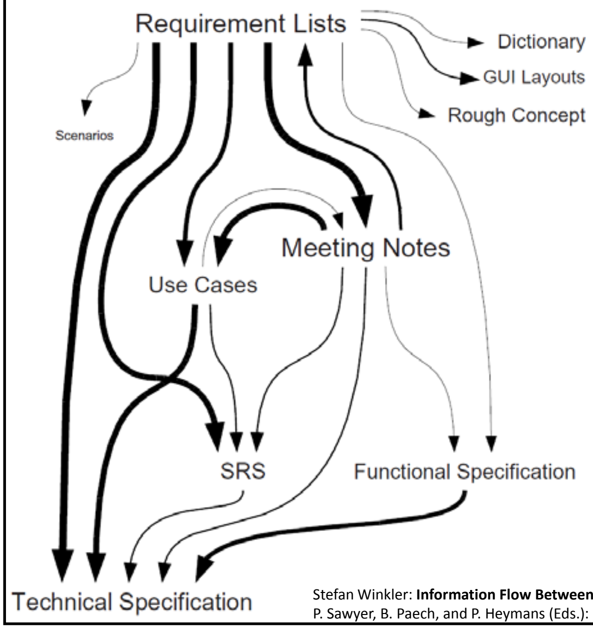
Concurrent Requirements Stores

- **In typical industrial settings, five to eight different media are used to store requirements.**



Stefan Winkler: **Information Flow Between Requirement Artifacts. Results of an Empirical Study**
P. Sawyer, B. Paech, and P. Heymans (Eds.): REFSQ 2007, LNCS 4542, pp. 232–246, Springer, 2007

Requirements Flow



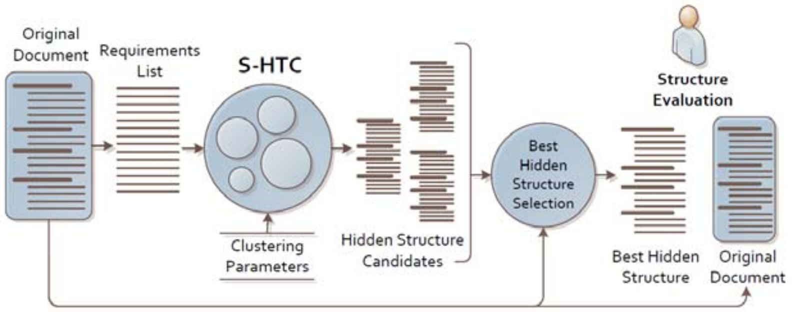
Stefan Winkler: **Information Flow Between Requirement Artifacts. Results of an Empirical Study**
P. Sawyer, B. Paech, and P. Heymans (Eds.): REFSQ 2007, LNCS 4542, pp. 232–246, Springer, 2007

Prof. Dr. Harald Störrle
Danmarks Tekniske Universitet (DTU)

Research

NL-Analysis of document outlines

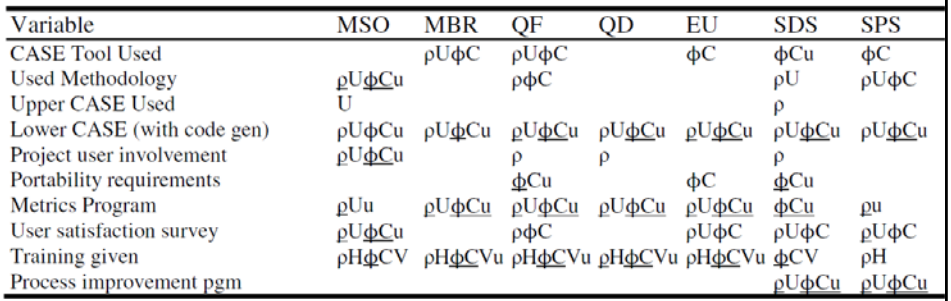
- There are algorithms to identify parts of requirements documents with poor structuring, sections that ought to be re-arranged, and requirements that are placed in conceptually unconnected sections.



Alessio Ferrari, Stefania Gnesi, Gabriele Tolomei: **Using Clustering to Improve the Structure of Natural Language Requirements Documents.** In: J. Doerr, A.L. Opdahl (Eds.): REFSQ 2013, LNCS 7830, pp. 34–49, 2013, Springer

Methods~Qualities (1/2)

- **Using the ISBSG data set on project outcomes, we can see some interesting correlations:**
  - Some methods/techniques have positive influence on many quality metrics.
  - Other actions seem to have little to no practical impact.
  - Some quality metrics are influenced positively by more or less any action.



Variable	MSO	MBR	QF	QD	EU	SDS	SPS
CASE Tool Used		ρUϕC	ρUϕC		ϕC	ϕCu	ϕC
Used Methodology	ρUϕCu		ρϕC			ρU	ρUϕC
Upper CASE Used	U					ρ	
Lower CASE (with code gen)	ρUϕCu	ρUϕCu	ρUϕCu	ρUϕCu	ρUϕCu	ρUϕCu	ρUϕCu
Project user involvement	ρUϕCu		ρ	ρ		ρ	
Portability requirements			ϕCu		ϕC	ϕCu	
Metrics Program	ρUu	ρUϕCu	ρUϕCu	ρUϕCu	ρUϕCu	ϕCu	ρu
User satisfaction survey	ρUϕCu		ρϕC		ρUϕC	ρUϕC	ρUϕC
Training given	ρHϕCV	ρHϕCVu	ρHϕCVu	ρHϕCVu	ρHϕCVu	ϕCV	ρH
Process improvement pgm						ρUϕCu	ρUϕCu

MSO: meet stated objectives	MBR: meet bus. Reqs.	QF: Quality of functionality
EU: Ease of use	SDS: speed of def. solution	SPS: speed of providing sol.

Łukasz Radliński: **Empirical Analysis of the Impact of Requirements Engineering on Software Quality**
B. Regnell and D. Damian (Eds.): REFSQ 2012, LNCS 7195, pp. 232–238, 2012, Springer

Methods~Qualities (2/2)

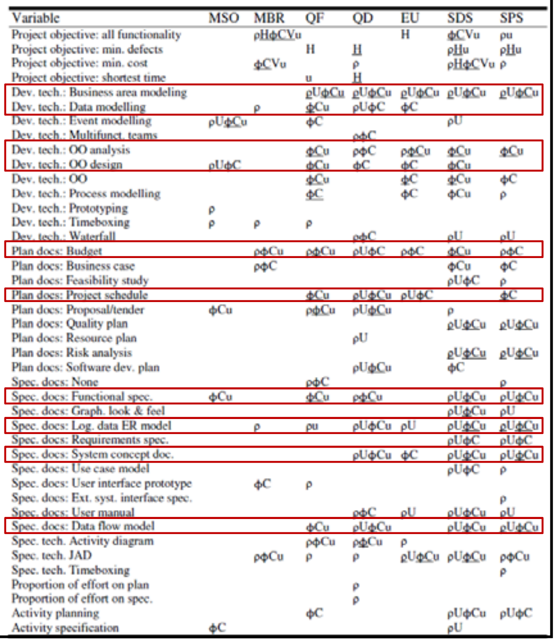
Dev. tech.: Business area modeling
Dev. tech.: Data modelling
Dev. tech.: OO analysis
Dev. tech.: OO design

Plan docs: Budget

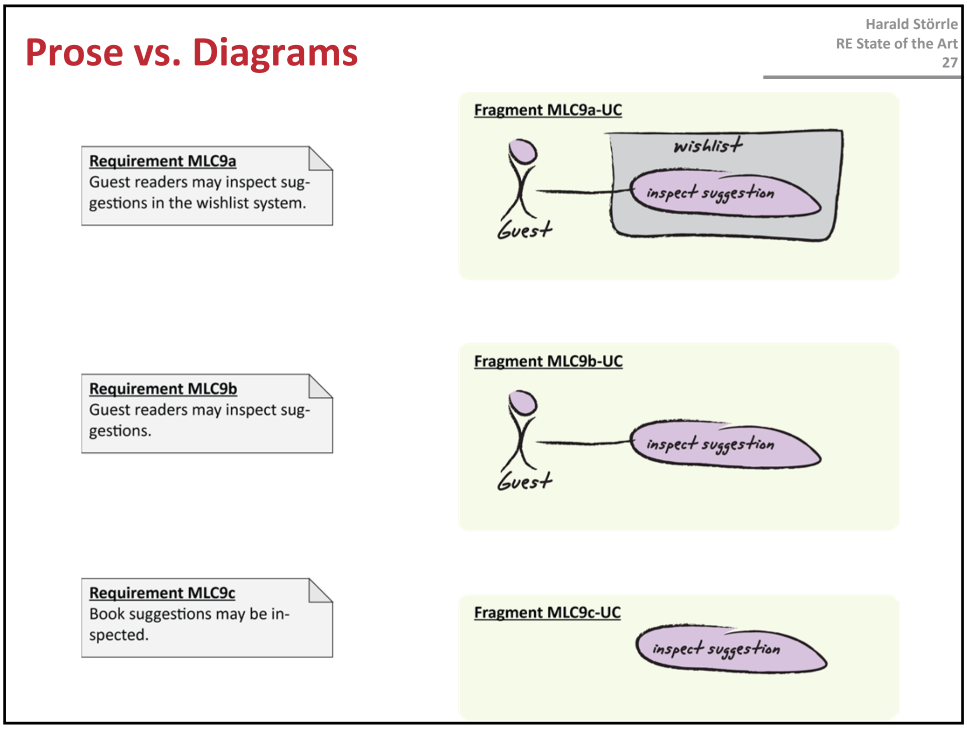
Plan docs: Project schedule

Spec. docs: Functional spec.
Spec. docs: Log. data ER model
Spec. docs: System concept doc.

Spec. docs: Data flow model



Variable	MSO	MBR	QF	QD	EU	SDS	SPS
Project objective: all functionality		ρHϕCVu			H	ϕCVu	ρu
Project objective: min. defects			H	H		ρHu	ρHu
Project objective: min. cost		ϕCVu		ρ		ρHϕCVu	ρ
Project objective: shortest time			u	H			
Dev. tech.: Business area modeling			ρUϕCu	ρUϕCu	ρUϕCu	ρUϕCu	ρUϕCu
Dev. tech.: Data modelling		ρ	ϕCu	ρUϕC	ϕC		
Dev. tech.: Event modelling	ρUϕCu		ϕC			ρU	
Dev. tech.: Multifunct. teams				ρϕC			
Dev. tech.: OO analysis			ϕCu	ρϕC	ρϕCu	ϕCu	ϕCu
Dev. tech.: OO design	ρUϕC		ϕCu	ϕC	ϕC	ϕCu	
Dev. tech.: OO			ϕCu		ϕC	ϕCu	ϕC
Dev. tech.: Process modelling			ϕC		ϕC	ϕCu	ρ
Dev. tech.: Prototyping	ρ						
Dev. tech.: Timeboxing	ρ	ρ	ρ				
Dev. tech.: Waterfall				ρϕC		ρU	ρU
Plan docs: Budget		ρϕCu	ρϕCu	ρUϕC	ρϕC	ϕCu	ρϕC
Plan docs: Business case		ρϕC				ϕCu	ϕC
Plan docs: Feasibility study						ρUϕC	ρ
Plan docs: Project schedule			ϕCu	ρUϕCu	ρUϕC		ϕC
Plan docs: Proposal/tender	ϕCu		ρϕCu	ρUϕCu		ρ	
Plan docs: Quality plan						ρUϕCu	ρUϕCu
Plan docs: Resource plan				ρU			
Plan docs: Risk analysis						ρUϕCu	ρUϕCu
Plan docs: Software dev. plan				ρUϕCu		ϕC	
Spec. docs: None			ρϕC				ρ
Spec. docs: Functional spec.	ϕCu		ϕCu	ρϕCu		ρUϕCu	ρUϕCu
Spec. docs: Graph. look & feel						ρUϕCu	ρU
Spec. docs: Log. data ER model		ρ	ρu	ρUϕCu	ρU	ρUϕCu	ρUϕCu
Spec. docs: Requirements spec.						ρUϕC	ρUϕC
Spec. docs: System concept doc.				ρUϕCu	ϕC	ρUϕCu	ρUϕCu
Spec. docs: Use case model						ρUϕC	ρ
Spec. docs: User interface prototype		ϕC	ρ				
Spec. docs: Ext. syst. interface spec.							ρ
Spec. docs: User manual				ρϕC	ρU	ρUϕCu	ρU
Spec. docs: Data flow model			ϕCu	ρUϕCu		ρUϕCu	ρUϕCu
Spec. tech. Activity diagram			ρϕCu	ρϕCu	ρ		
Spec. tech. JAD		ρϕCu	ρ	ρ	ρUϕCu	ρUϕCu	ρϕCu
Spec. tech. Timeboxing							ρ
Proportion of effort on plan				ρ			
Proportion of effort on spec.				ρ			
Activity planning			ϕC			ρUϕCu	ρUϕC
Activity specification	ϕC					ρU	



Prose vs. Diagrams

Requirement MLC9a
Guest readers may inspect suggestions in the wishlist system.

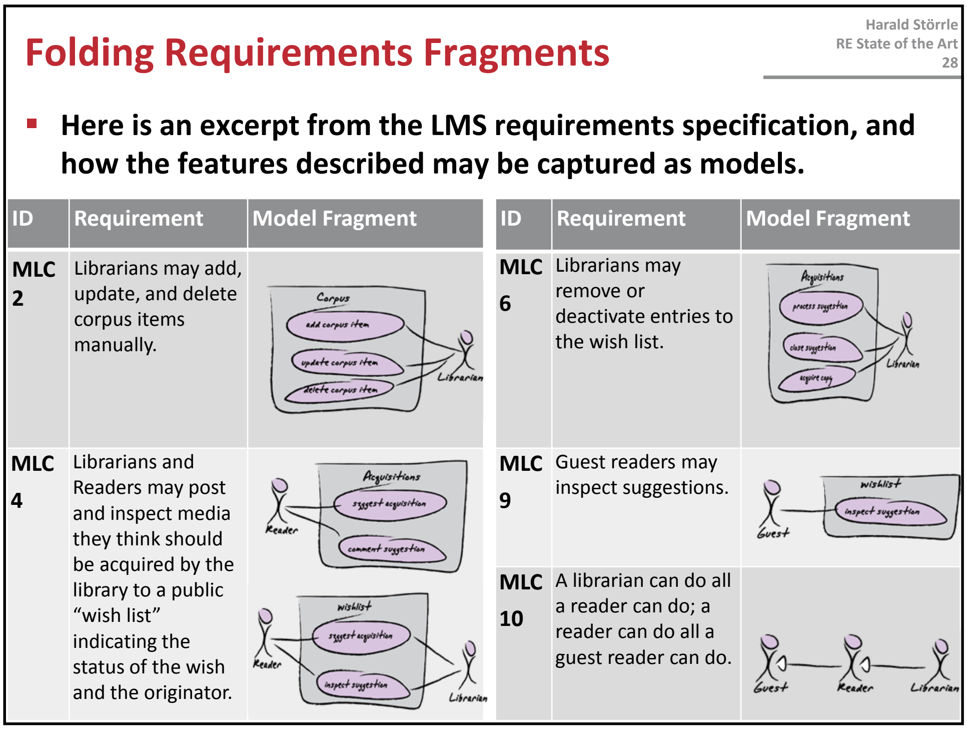
Fragment MLC9a-UC

Requirement MLC9b
Guest readers may inspect suggestions.

Fragment MLC9b-UC

Requirement MLC9c
Book suggestions may be inspected.

Fragment MLC9c-UC



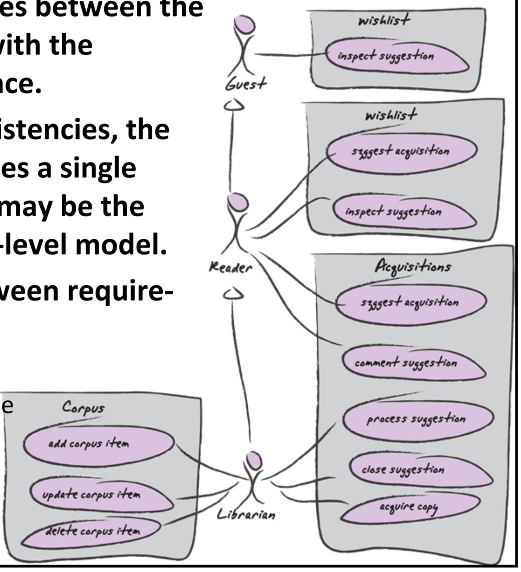
Folding Requirements Fragments

- **Here is an excerpt from the LMS requirements specification, and how the features described may be captured as models.**

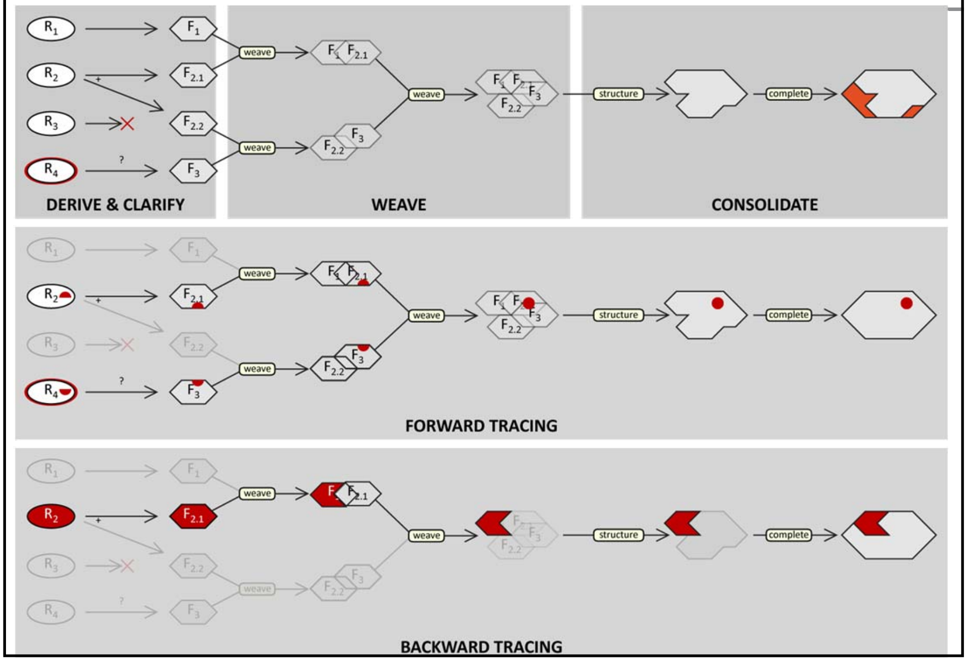
ID	Requirement	Model Fragment
MLC 2	Librarians may add, update, and delete corpus items manually.	Corpus add corpus item update corpus item delete corpus item Librarian
MLC 4	Librarians and Readers may post and inspect media they think should be acquired by the library to a public “wish list” indicating the status of the wish and the originator.	Acquisitions suggest acquisition comment suggestion Reader wishlist suggest acquisition inspect suggestion Reader Librarian
MLC 6	Librarians may remove or deactivate entries to the wish list.	Acquisitions process suggestion close suggestion acquire copy Librarian
MLC 9	Guest readers may inspect suggestions.	
MLC 10	A librarian can do all a reader can do; a reader can do all a guest reader can do.	

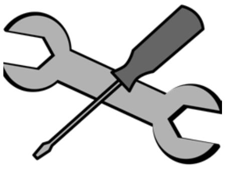
Folding Requirements Fragments

- **Folding these Use Case Diagrams detects certain kinds of inconsistencies between the requirements, thus helping with the requirements quality assurance.**
- **If there are no (more) inconsistencies, the folding succeeds and generates a single large Use Case Diagram that may be the starting point of our analysis-level model.**
- **Folding establishes links between requirements and model elements.**
  - Questions about one of them may be answered with recourse to the other.
  - The impact of changes to one side can easily be traced to the other side.



Fragment Weaving



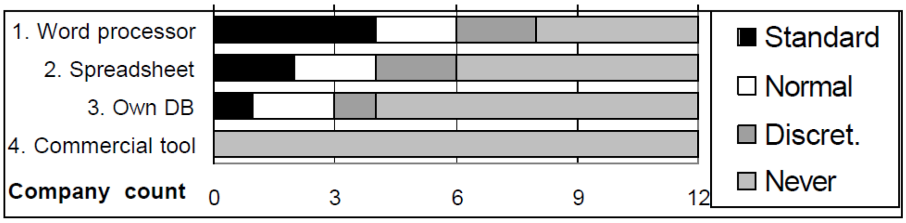


Prof. Dr. Harald Störrle
Danmarks Tekniske Universitet (DTU)

Tools

Tools

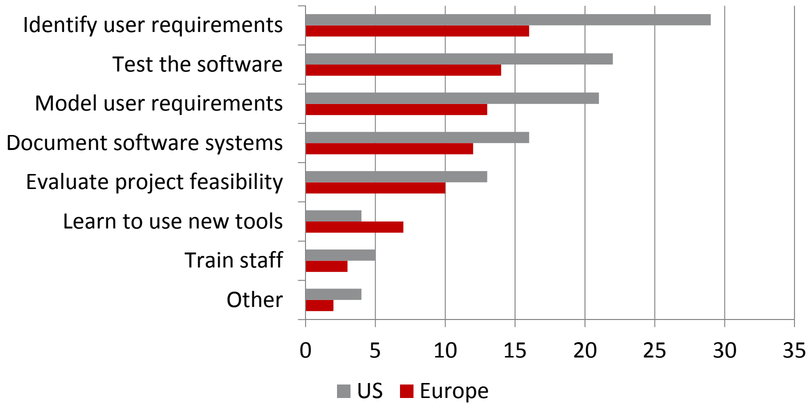
Do you use any tool supporting requirements analysis and top-level design?	How many employees and consultants are there in your company?				
	1–5	6–20	21–50	51–100	More than 100
Yes	16%	18%	33%	33%	51%
No	84%	82%	67%	67%	49%



Luisa Mich, Mariangela Franch, Pierluigi Novi Inverardi: **Market research for requirements analysis using linguistic tools**
Requirements Eng (2004) 9: 40–56, Springer

Tools

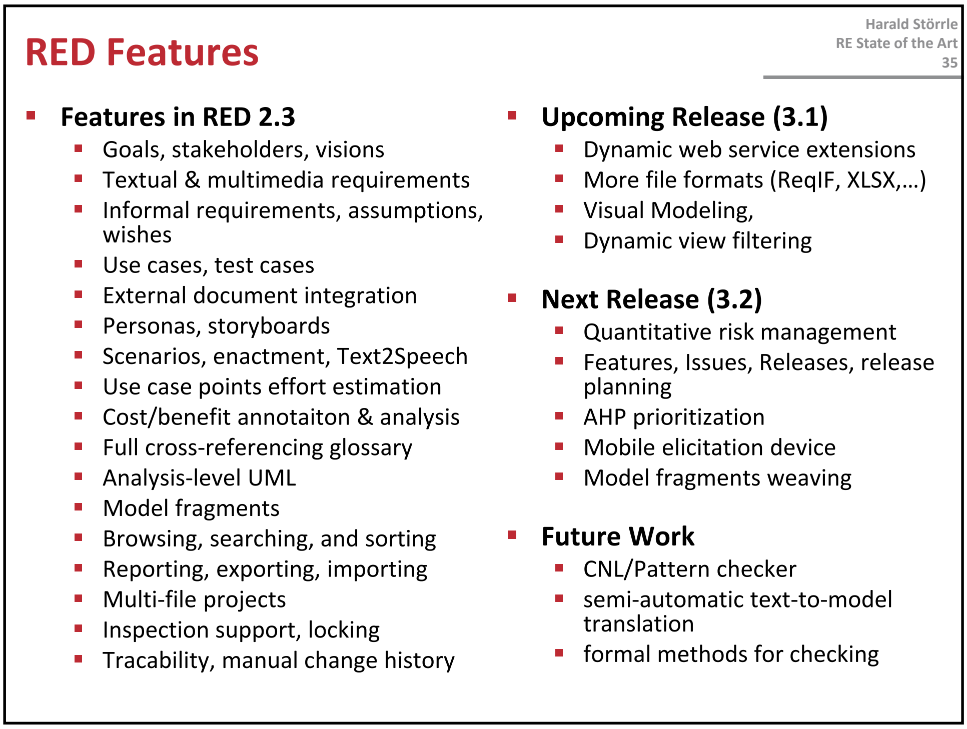
- ‘‘I hate to be a cynic, but there are hardly any worthwhile tools. The overhead in learning to use them is too great for the payoff.’’



Luisa Mich, Mariangela Franch, Pierluigi Novi Inverardi: **Market research for requirements analysis using linguistic tools**
Requirements Eng (2004) 9: 40–56, Springer

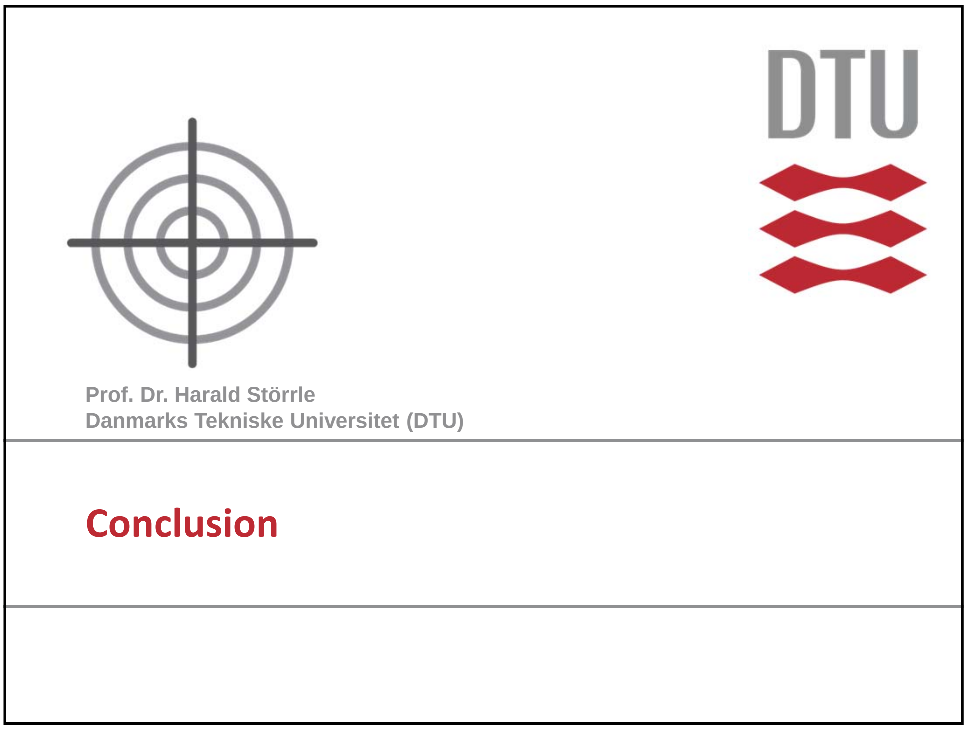
The Requirements Editor RED

- **RED is a stand-alone tool for requirements engineering developed by students.**
  - Development has started in September 2011, a major re-engineering took place in 2013.
  - Altogether, 7 MSc theses have been invested into RED, 4 more are currently.
  - A student helper has been employed as Build Manager since late 2013 to coordinate and test the student’s contributions, find and eliminate bugs, and apply small improvements continuously.
- **RED is intended as a tool to support teaching, it’s development aspires to optimize conceptual clarity and coherence, while offering a comprehensive and practical toolbox.**
  - Based on Eclipse RCP, it’s infratructure is generated from a meta-model.



RED Features

- **Features in RED 2.3**
  - Goals, stakeholders, visions
  - Textual & multimedia requirements
  - Informal requirements, assumptions, wishes
  - Use cases, test cases
  - External document integration
  - Personas, storyboards
  - Scenarios, enactment, Text2Speech
  - Use case points effort estimation
  - Cost/benefit annotaiton & analysis
  - Full cross-referencing glossary
  - Analysis-level UML
  - Model fragments
  - Browsing, searching, and sorting
  - Reporting, exporting, importing
  - Multi-file projects
  - Inspection support, locking
  - Tracability, manual change history
- **Upcoming Release (3.1)**
  - Dynamic web service extensions
  - More file formats (ReqIF, XLSX,…)
  - Visual Modeling,
  - Dynamic view filtering
- **Next Release (3.2)**
  - Quantitative risk management
  - Features, Issues, Releases, release planning
  - AHP prioritization
  - Mobile elicitation device
  - Model fragments weaving
- **Future Work**
  - CNL/Pattern checker
  - semi-automatic text-to-model translation
  - formal methods for checking



Prof. Dr. Harald Störrle
Danmarks Tekniske Universitet (DTU)

Conclusion

How to improve?

Anecdotes are not scientific evidence

STRIBEN
af
WulffMorgenthaler
Den højt respekterede professor Nibbowitz beviser, at blæksprutter er mere intelligente end katte, når de udsættes for samme udfordringer under samme betingelser

The highly respected Professor Nibbowitz proved, that octopus are more intelligent than cat, when exposed to the same challenges and conditions.

How to improve?

- **Educate future Requirements Engineers well**
  - Establish RE in curriculum
  - Teach mix of pragmatic topics
- **Select relevant research topics**
  - Minimal increments to exotic topics are hardly relevant
  - Empirical results add to the body of knowledge
- **Implement**
  - Provide implementation of advanced ideas/concepts
- **Stay informed**
  - Allow interested individuals to maintain academic interest
  - Use results where available
- **Investigate**
  - Conduct your own research
  - Collect data
  - Try and educate your clients
- **Talk**
  - Allow researchers in to
  - Report on experiences

The Model Observatory

Why do people model and how do they use their models?
Are there any differences between different groups?
Does it pay to model, and if so: when and why?

Help us answer these questions and more by answering a few questions - it takes less than 5 minutes!

http://tinyurl.com/MU-survey-2014

Prof. Dr. Harald Störrle
Department of Applied Mathematics and Computer Science
Technical University of Denmark
Matematiktorvet
Building 303b, Room 056
DK-2800 Kgs. Lyngby

hsto@dtu.dk
www.compute.dtu.dk/~hsto

DTU