

Multimedia Didactics for Software Engineering

Ralf S. Engelschall

2023-07-31, Helmut-Balzert-Preis, Jury-Presentation



**Software
Engineering
Academy**



About Author

- **Dr. Ralf S. Engelschall** (50)
- Science Context:
 - Computer Science Study at *TU München*
 - Computer Science Doctorate at *Uni Augsburg*
- Work Roles:
 - Director of *msg Research*
 - Director of *Software Engineering Academy*
 - Lecturer at *TU München*
- Education Activities:
 - *Software Engineering in der industriellen Praxis (SEIP)*
 - *Software Engineering Fundamentals (SEF)*
 - *Software Architecture Fundamentals (SAF)*
 - *IT-Architekten-Ausbildungsprogramm (ITAAP)*
 - *Trend Competence Program (TCP)*
 - *Container-Based Deployment (CBD)*



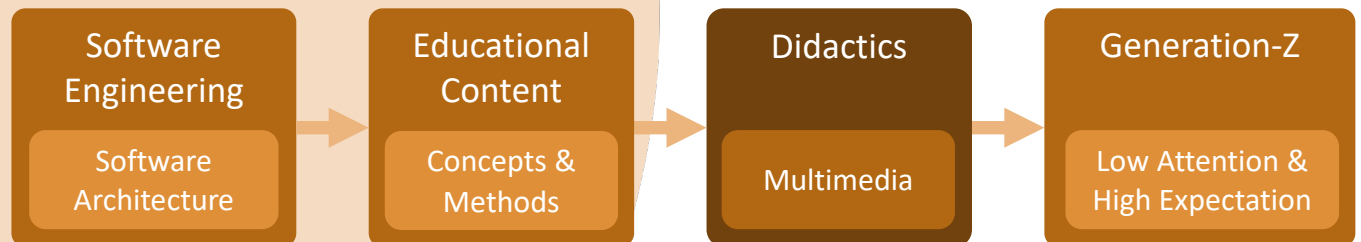
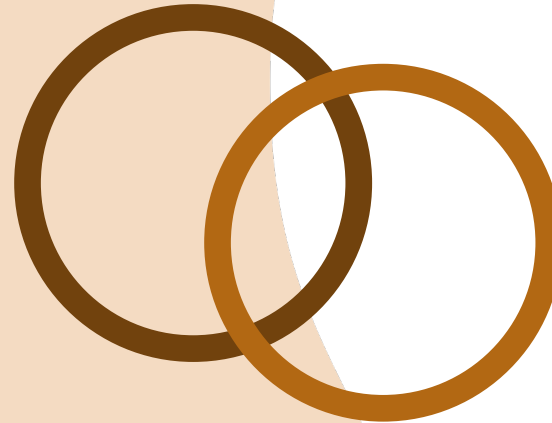
My Education Challenges

- **Different Education Contexts**
university lectures, industry trainings,
different durations/brandings, etc
- **Unsexy and Complex Topic**
non-agile upfront thinking,
many Software Engineering disciplines, etc.
- **Generation-Z Low Attention Threshold**
social acceleration, information overflow,
strong catchyness expectation, etc.
- **Post-Pandemic New Work**
online-first, home-office-driven,
reduced traveling (sustainability), etc.



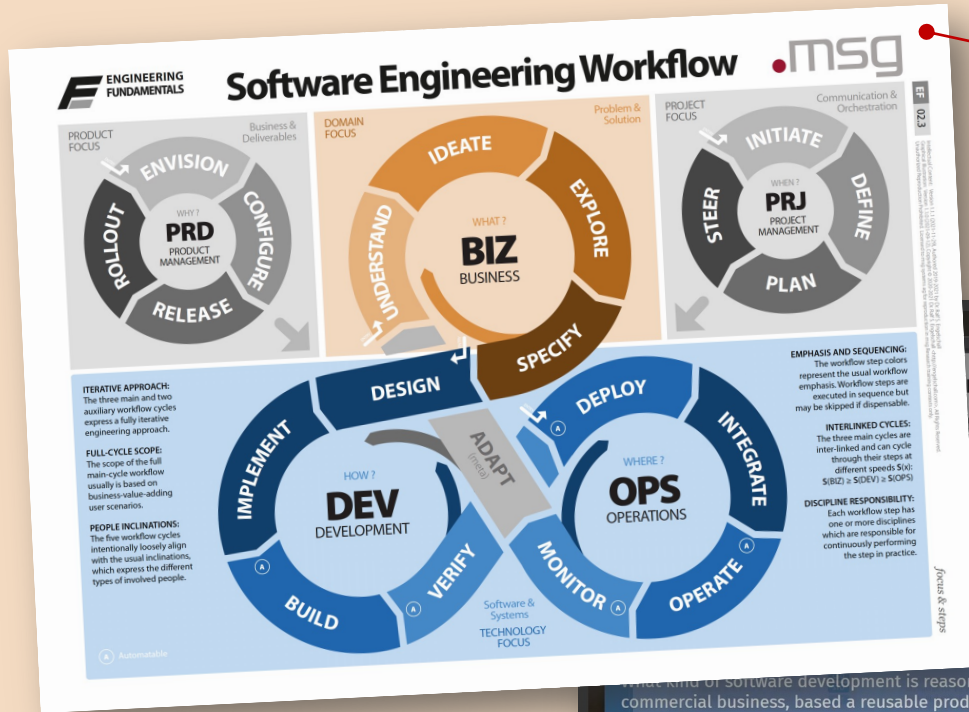
My Education Context

- **Didactics:**
the approaches, methods and activities of systematic teaching.
- **Multimedia:**
the form of communication that combines different content forms (text, images, video, audio).



My Education Solution

Multimedia Didactics for Software Engineering



Remember: "multimedia", in this context, primarily refers to the educational content and its various formats (poster, handout, canvas, etc), variants (branding, annotation, etc), combinations of content (image, speech, etc) and channels (print, video, etc), and not just the use of a video-stream!

Classes TUM TECHNISCHE UNIVERSITÄT MÜNCHEN

Open Source Software Development - OSS
Main commercial development of standard, highly customizable, and fully reusable generic software for a class of customers.

Software Development Kit - SDK
Software libraries and frameworks of reusable functionality for developing software.
Examples: iOS SDK, XAPK, Golang, Qt, ROS, Scikit-Learn, J2EE, Spring, Hibernate, etc.

Software Development Tools
Software tools for editing, linking, compiling, packaging, distributing, and installing software.

Willkommen zu SEIP!

VOTE (temporarily disabled until new voting starts)

1	2	3
4	5	6
7	8	9
YES	NO	Abstain

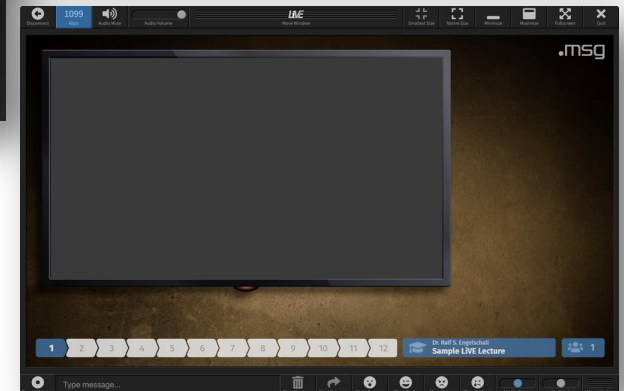
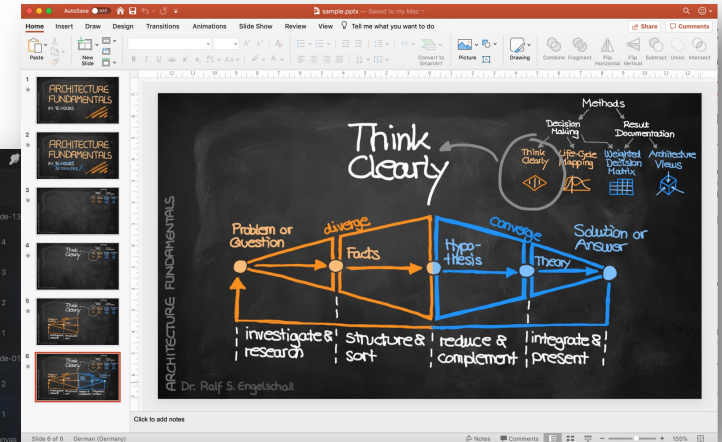
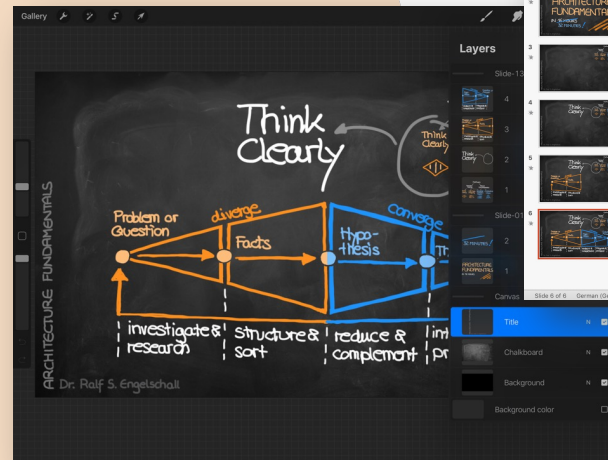
FEEDBACK: Consent, Refusal, Surprise, Smile, Frown, Sadness

FEELING: sub, CHALLENGE, over, tired, MOOD, excited

Dr. Ralf S. Engelschall
TUM-SEIP 01: Software Engineering

My Didactics Journey

- ...many experiments...
- 2019: Hand-Drawn Slides (<https://github.com/rse/psd2pptx>) due to non-vector format, does not allow flexible content zooming and annotation.
- 2020: Desktop Player App (<https://github.com/rse/live-receiver>) download and executable signing causes too high entry barrier.
- ...and even more...



My Education Building Blocks

12 essential building blocks for a really strong overall lecture experience



What-Why-Where Paradigm



Online-First Approach



Gamification Quizzes



Practice Back-Pressure



Blended Studio Reality



Live Attendee Feedback



Diagram-Driven Education



Content Zooming and Annotation



Perception Amplifier



Generative Lecture Content



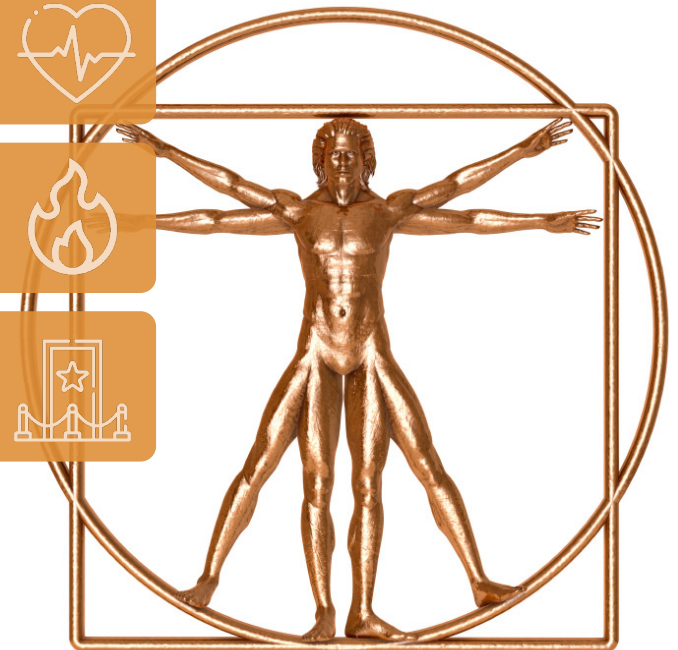
Live Questions and Answers



Backstage Pass



Didactics Crux and Game Changer



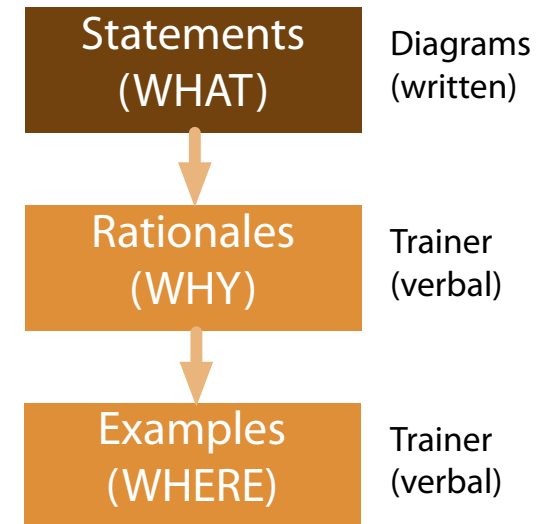
What-Why-Where Paradigm



Building Block 1/12

- **WHAT:**
show the *What* on written material,
vocally explain the *Why*,
vocally tell about the practical *Where*.
- **WHY (DIDACTICS):**
the *What* has to be remembered,
the *Why* is the key for acceptance,
the *Where* is the example to understand it.
- **WHY (OTHER):**
students still need both the content
(written) and lecturer (vocally), so
still attend the lecture.

Didactics Crux!

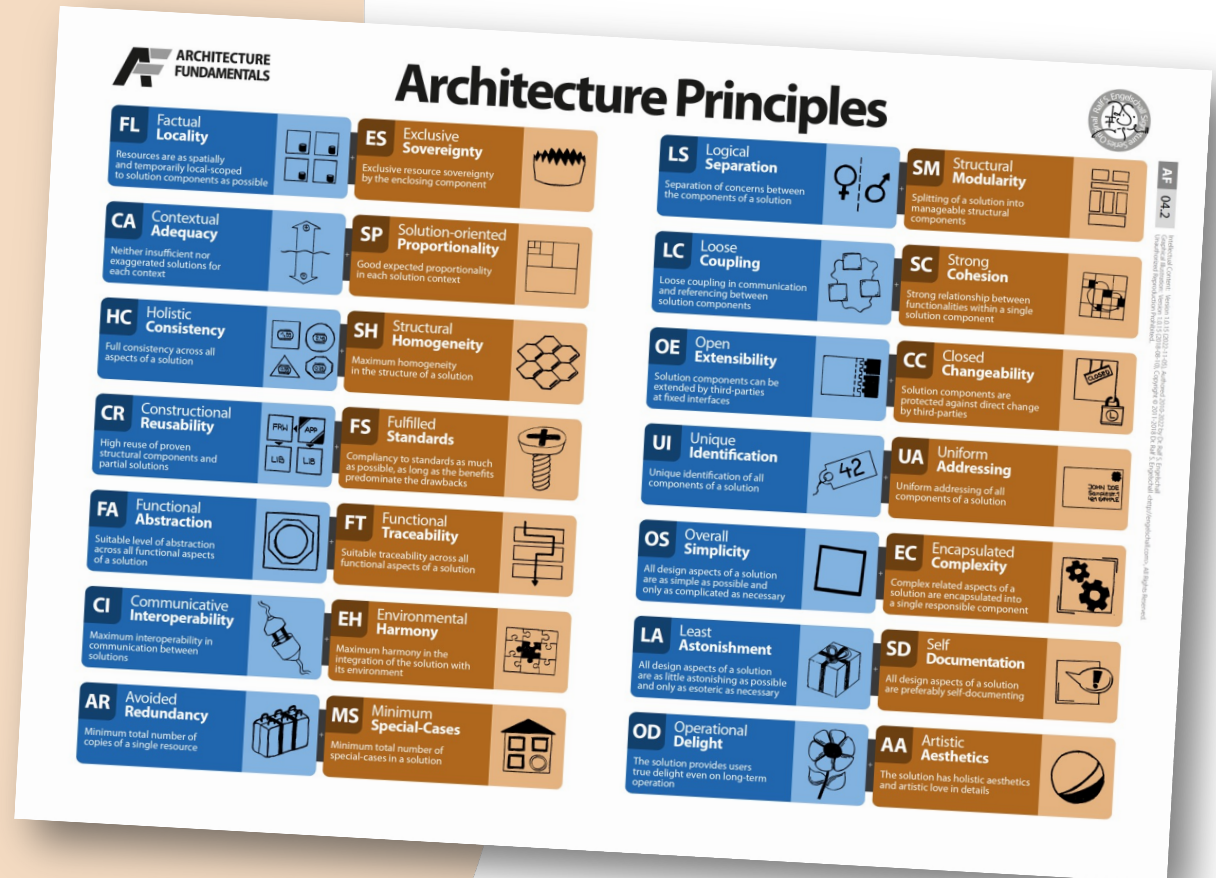


Practice Back-Pressure



Building Block 2/12

- **WHAT:**
provide *What* artifacts for practice, use them 1:1 also for education, minimize education-only artifacts.
- **WHY (DIDACTICS):**
avoid mental gaps between education and subsequent practice, and practice relevance always convinces.
- **WHY (OTHER):**
(none)



ARCHITECTURE FUNDAMENTALS

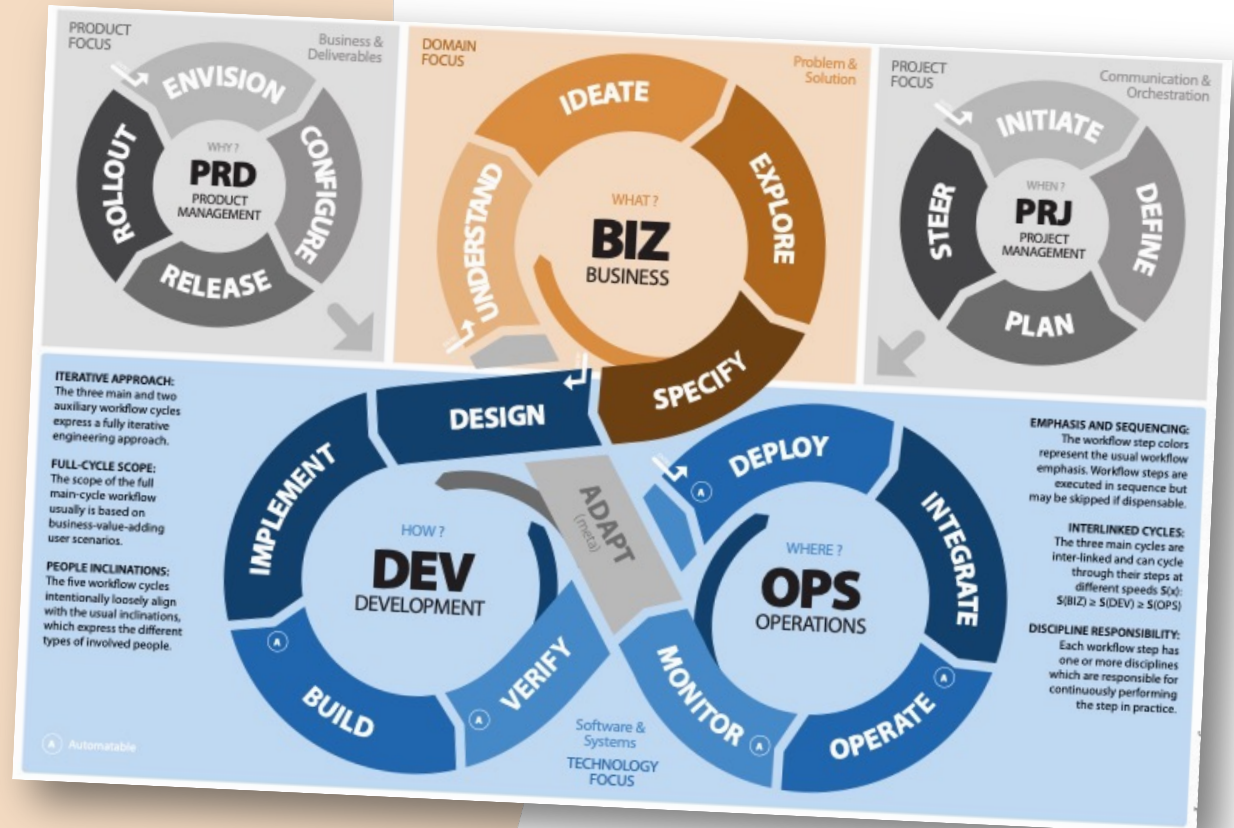
Architecture Principles

AR 042

FL Factual Locality Resources are as spatially and temporally local-stopped to solution components as possible	ES Exclusive Sovereignty Exclusive resource sovereignty by the enclosing component	LS Logical Separation Separation of concerns between the components of a solution	SM Structural Modularity Splitting of a solution into manageable structural components
CA Contextual Adequacy Neither insufficient nor exaggerated solutions for each context	SP Solution-oriented Proportionality Good expected proportionality in each solution context	LC Loose Coupling Loose coupling in communication and referencing between solution components	SC Strong Cohesion Strong relationship between functionalities within a single solution component
HC Holistic Consistency Full consistency across all aspects of a solution	SH Structural Homogeneity Maximum homogeneity in the structure of a solution	OE Open Extensibility Solution components can be extended by third parties at fixed interfaces	CC Closed Changeability Solution components are protected against direct change by third parties
CR Constructional Reusability High reuse of proven structural components and partial solutions	FS Fulfilled Standards Compliance to standards as much as possible, as long as the benefits predominate the drawbacks	UI Unique Identification Unique identification of all components of a solution	UA Uniform Addressing Uniform addressing of all components of a solution
FA Functional Abstraction Suitable level of abstraction across all functional aspects of a solution	FT Functional Traceability Suitable traceability across all functional aspects of a solution	OS Overall Simplicity All design aspects of a solution are as simple as possible and only as complicated as necessary	EC Encapsulated Complexity Complex related aspects of a solution are encapsulated into a single responsible component
CI Communicative Interoperability Maximum interoperability in communication between solutions	EH Environmental Harmony Maximum harmony in the integration of the solution with its environment	LA Least Astonishment All design aspects of a solution are as little astonishing as possible and only as esoteric as necessary	SD Self Documentation All design aspects of a solution are preferably self-documenting
AR Avoided Redundancy Minimum total number of copies of a single resource	MS Minimum Special-Cases Minimum total number of special-cases in a solution	OD Operational Delight The solution provides users true delight even on long-term operation	AA Artistic Aesthetics The solution has holistic aesthetics and artistic love in details



- **WHAT:** describe complex aspects, concepts or methods through visual models on dedicated diagrams.
- **WHY (DIDACTICS):** precision through abstraction and conciseness, and easier comprehension of reality.
- **WHY (OTHER):** visibility, recognition, catchyness.

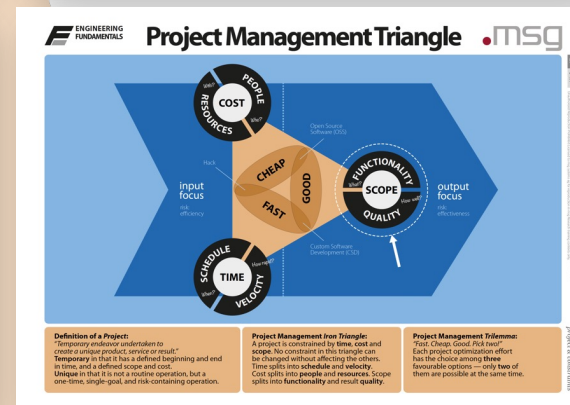
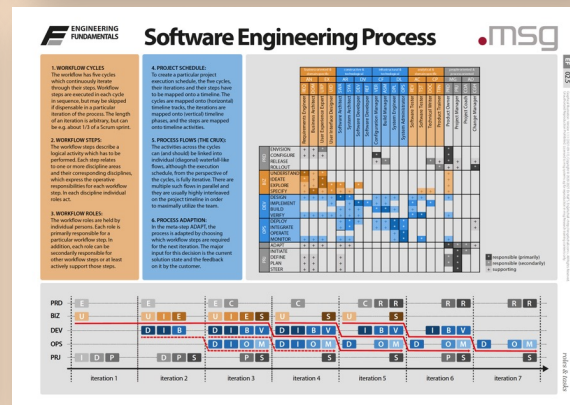
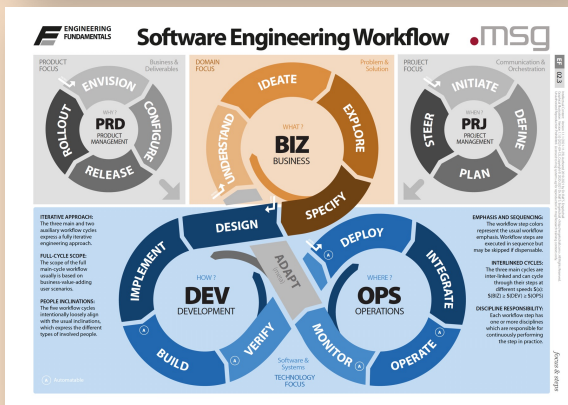
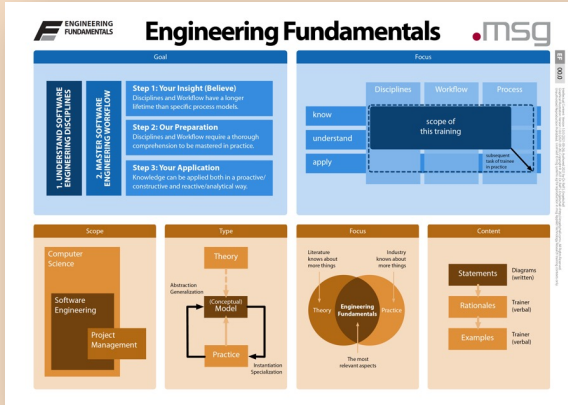


Didactics Crux!

Diagram-Driven Education



Building Block 3/12

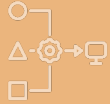




- WHAT:**
 select arbitrary combination of diagrams, overlay with logo and hints (duration, quiz markers), and generate multiple artifacts (agenda, content, handout, etc).
- WHY (DIDACTICS):**
 different content orders,
 different target groups.
- WHY (OTHER):**
 different artifacts,
 different lecture times,
 different lecture formats.

	A	B	K	M	N	O	P	AF	AO	AP	AQ	AR	AS	AT	AU	AV	AW
1			msg-gdita		msg-itaap-m1	msg-itaap-m2		msg-sef		tum-seip-01	tum-seip-02	tum-seip-03	tum-seip-04	tum-seip-05	tum-seip-06	tum-seip-07	tum-seip-08
2			msg		msg	msg		msg		tum	tum	tum	tum	tum	tum	tum	tum
3			16		16	8,00		8,00		SE	SE	AR	AR	AR	AR	SCM	PM
4	SOLL		960		960	480		480		180	180	180	180	180	180	180	180
5	IST		955		960	480		480		180	180	180	180	180	180	180	180
19	ENGINEERING FUNDAMENTALS																
20	EF-01.1	Software Classes						25		25							
21	EF-01.2	Software Development Approaches						30		25							
22	EF-01.3	Software Engineering						15		15							
23	EF-01.4	TRUE Manifesto						15		20							
24	EF-01.5	Sustainability						10									
25	EF-02.1	Software Engineering Meta-Model						25		25							
26	EF-02.2	Software Engineering Disciplines	1		5			30		40							
27	EF-02.3	Software Engineering Workflow	1		5			35			25						
28	EF-02.4	Software Engineering Steps						5		5							
29	EF-02.5	Software Engineering Process						30		30							
30	EF-02.6	Software Engineering Artifacts						30		30							
31	EF-03.1	Software Engineering Efforts						15		15							
32	EF-03.2	Software Engineering Uncertainty						10		15							
33	EF-03.3	Software Engineering Estimation						15									
34	EF-04.1	Project Management Triangle						10									15
35	EF-04.2	Project Management Building Blocks						25									30
36	EF-04.3	Project Management Plan-Driven															20
37	EF-04.4	Project Management Agile															20
38	CONTEXT & ENTITLEMENT																
39	AF-01.1	Architecture Stargate	15		20							30					
40	AF-01.2	Adequacy and Beauty	10		10							10					
41	AF-01.3	King Discipline Architecture	3		5							10					
42	AF-01.4	Architecture Manifesto	10		10							10					
43	AF-01.5	Complex vs. Complicated	5		5							5					
44	SPACE & ONTOLOGY																
45	AF-02.1	Architecture Space	20		20							20					
46	AF-02.2	Architecture Ontology	20		20							20					

Generative Lecture Content



Building Block 4/12



TUM WS22/23, SEIP Modul 03 (2022-11-03)

Begin	End	Duration	Topic
14:00	14:30	00:30	AF-01.1 Architecture Stargate
14:30	14:40	00:10	AF-01.2 Adequacy and Beauty
14:40	14:50	00:10	AF-01.3 King Discipline Architecture
14:50	15:00	00:10	AF-01.4 Architecture Manifesto
15:00	15:05	00:05	AF-01.5 Complex vs. Complicated
15:05	15:25	00:20	AF-02.1 Architecture Space
15:25	15:40	00:15	Break
15:40	16:00	00:20	AF-02.2 Architecture Ontology
16:00	17:00	01:00	AF-04.1 Architecture Maxims

ITAAP Modul 1

Begin	End	Duration	Topic
09:00	09:20	00:20	Introduction Round
09:20	09:30	00:10	AF-00.0 Architecture Fundamentals
09:30	09:35	00:05	EF-02.2 Software Engineering Disciplines
09:35	09:40	00:05	EF-02.3 Software Engineering Workflow
09:40	10:00	00:20	AF-01.1 Architecture Stargate
10:00	10:10	00:10	AF-01.2 Adequacy and Beauty
10:10	10:15	00:05	AF-01.3 King Discipline Architecture
10:15	10:25	00:10	AF-01.4 Architecture Manifesto
10:25	10:30	00:05	AF-01.5 Complex vs. Complicated
10:30	10:40	00:10	Break
10:40	11:00	00:20	AF-02.1 Architecture Space
11:00	11:20	00:20	AF-02.2 Architecture Ontology
11:20	11:25	00:05	AF-03.1 Requirements Basics
11:25	12:10	00:45	AF-04.1 Architecture Maxims
12:10	12:55	00:45	Lunch
12:55	13:55	01:00	AF-04.2 Architecture Principles
13:55	14:25	00:30	AF-05.1 Component Design
14:25	14:35	00:10	Break
14:35	15:25	00:50	AF-05.2 Interface Design
15:25	15:40	00:15	AF-05.3 Component Hierarchy
15:40	15:50	00:10	Break
15:50	16:10	00:20	AF-06.2 Layer Architecture
16:10	16:20	00:10	Break
16:20	16:40	00:20	AF-06.3 Slice Architectures
16:40	16:55	00:15	AF-07.1 Flow Architectures
16:55	17:10	00:15	AF-07.2 Process Architectures

Begin	End	Duration	Topic
09:00	09:05	00:05	Recap
09:05	09:25	00:20	AF-07.3 Cluster Architectures
09:25	09:45	00:20	AF-08.1 Networking Architectures
09:45	10:05	00:20	AF-08.2 Communication Architectures
10:05	10:45	00:40	AF-09.1 Data Structure Architectures
10:45	10:55	00:10	Break
10:55	11:20	00:25	AF-10.1 Application Reference Architecture
11:20	11:35	00:15	AF-10.2 Client-Server Architecture
11:35	12:20	00:45	AF-10.3 Information System Architecture
12:20	13:05	00:45	Lunch
13:05	13:25	00:20	AF-10.4 Reactive System Architecture
13:25	13:35	00:10	AF-11.3 Application Composition
13:35	13:45	00:10	AF-16.1 Think Clearly
13:45	13:55	00:10	AF-16.3 Technology Life-Cycles
13:55	14:10	00:15	AF-16.4 Open Source Software
14:10	14:25	00:15	AF-16.5 Back of the Envelope Calculation
14:25	14:35	00:10	Break
14:35	14:50	00:15	AF-16.6 Weighted Decision Matrix
14:50	15:00	00:10	AF-16.7 Focus Area Maturity Model
15:00	15:20	00:20	AF-17.1 Big Picture
15:20	15:40	00:20	AF-17.2 Viewpoints & Perspectives
15:40	16:05	00:25	AF-Practice: Viewpoints & Perspectives
16:05	16:10	00:05	AF-19.0 Bibliography
16:10	16:30	00:20	Summary
16:30	16:50	00:20	Feedback Round



The image shows three overlapping slide thumbnails for a presentation titled "Architecture Stargate". Each slide features the "ARCHITECTURE FUNDAMENTALS" logo in the top left corner. The main content area is divided into three sections: "Requirements Process Estimation" (light blue), "Engineering" (dark blue), and "Craftsmanship" (orange). A vertical red bar with the numbers "1" and "2" is positioned on the left side of each slide. On the right side, there are logos for "MSG" (top), "TUM TECHNISCHE UNIVERSITÄT MÜNCHEN" (middle), and "MSG" (bottom). A vertical red bar on the right side of the slides contains the text "AF 01.1" and "Individual Content". The top slide has a page number "20" in the top right corner, and the middle slide has a page number "30" in the top right corner.

Generative Lecture Content



Building Block 4/12



Zusammenfassung

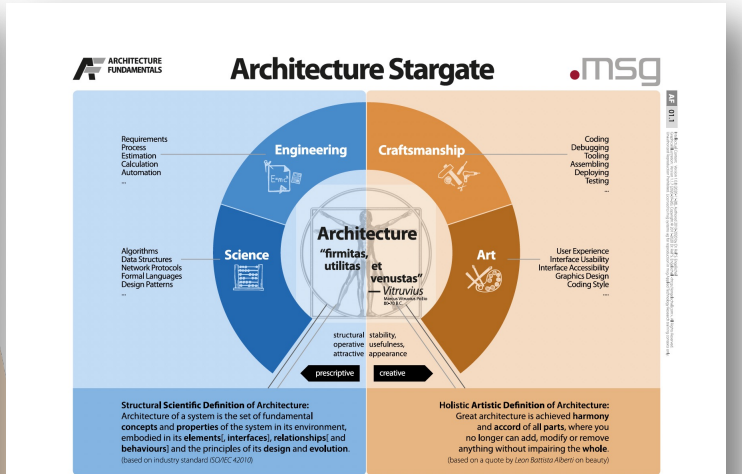
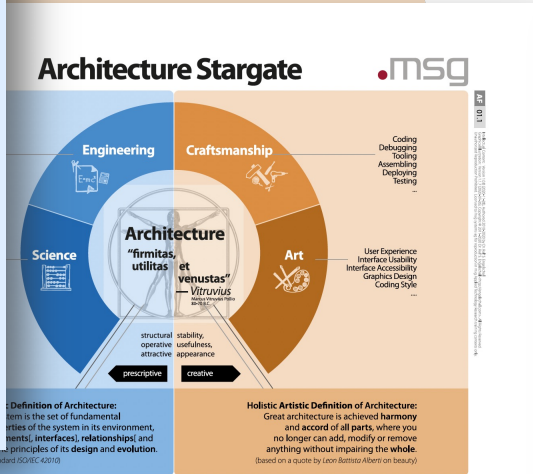
Architektur ist nicht einfach zu definieren. Man kann Architektur sowohl ***strukturell wissenschaftlich*** über messbare Elemente (elements), Schnittstellen (interfaces) und Beziehungen (relationships), als auch ***ganzheitlich künstlerisch*** über "die Harmonie und den Einklang aller Teile" definieren. Die "Wahrheit" liegt in der Praxis irgendwo dazwischen, denn die beiden Extrema spannen einen breiten Raum auf, in dem alle Lösungen in der Praxis liegen.

Auf der strukturell wissenschaftlichen Seite definiert sich Architektur über die Aspekte ***Science*** (insbesondere Computer Science) und ***Engineering*** (insbesondere Software Engineering). Auf der ganzheitlich künstlerischen Seite definiert sich Architektur über die Aspekte ***Craftsmanship*** (Handwerk, insbesondere Programmieren) und ***Art*** (Kunst, insbesondere User Experience).

Summary

Architecture is not easy to define. You can define architecture both ***structurally scientific*** through measurable elements, interfaces and relationships, or also ***wholly artistic*** through "the harmony and the accord of all parts." The "truth" lies somewhere in practice in between, because the two extremes span a broad space, in which all solutions are located in practice.

On the structurally scientific side, architecture defines itself through the aspects ***Science*** (in particular Computer Science) and ***Engineering*** (especially Software Engineering). On the holistic artistic side, architecture defines itself through the aspects of ***Craftsmanship*** (especially programming) and ***Art*** (especially User Experience).



Architektur ist nicht einfach zu definieren. Man kann Architektur sowohl **strukturell wissenschaftlich** über messbare Elemente (elements), Schnittstellen (interfaces) und Beziehungen (relationships), als auch **ganzheitlich künstlerisch** über "die Harmonie und den Einklang aller Teile" definieren. Die "Wahrheit" liegt in der Praxis irgendwo dazwischen, denn die beiden Extrema spannen einen breiten Raum auf, in dem alle Lösungen in der Praxis liegen.

Auf der strukturell wissenschaftlichen Seite definiert sich Architektur über die Aspekte **Science** (insbesondere Computer Science) und **Engineering** (insbesondere Software Engineering). Auf der ganzheitlich künstlerischen Seite definiert sich Architektur über die Aspekte **Craftsmanship** (Handwerk, insbesondere Programmieren) und **Art** (Kunst, insbesondere User Experience).

Architecture is not easy to define. You can define architecture both **structurally scientific** through measurable elements, interfaces and relationships, or also **wholly artistic** through "the harmony and the accord of all parts." The "truth" lies somewhere in practice in between, because the two extremes span a broad space, in which all solutions are located in practice.

On the structurally scientific side, architecture defines itself through the aspects **Science** (in particular Computer Science) and **Engineering** (especially Software Engineering). On the holistic artistic side, architecture defines itself through the aspects of **Craftsmanship** (especially programming) and **Art** (especially User Experience).

Online-First Approach



Building Block 5/12

- **WHAT:**
design lecture setup primarily for online performance, use reduced setup for on-site performance.
- **WHY (DIDACTICS):**
Generation-Z expectations (*YouTube/TikTok*-style consumption), post-pandemic *New Work* era, fresher attendees.
- **WHY (OTHER):**
(none)



TUM Lehrveranstaltung
SOFTWARE ENGINEERING IN DER INDUSTRIELLEN PRAXIS (SEIP)
(IN2235) im WS22/23, in Kooperation von

TUM TECHNISCHE UNIVERSITÄT MÜNCHEN
Dr. Prof. Florian Matthes
sebis Lehrstuhl, TUM
(Verantwortung, Organisation, Klausur)

Software Engineering Academy
Dr. Ralf S. Engelschall
SEA Software Engineering Academy gGmbH
(Lehrauftrag, Materialien, Infrastruktur)

msg
Dr. Ralf S. Engelschall
msg Research, msg systems ag
(Lehrauftrag, Kontext, Erfahrungen)

Software Classes

What kind of software development is reasonable for establishing a commercial business, based a reusable product?

1: Custom Software Development	2
2: Standard Software Development	14/18 (78%)
3: Open Source Software	1

Hubs Pad
MESSAGE
Willkommen zu SEIP!
Clear Send
VOTE: Temporarily disabled until new voting starts
1 2 3
4 5 6
7 8 9
YES NO Abstain
FEEDBACK
Globe Message Like Dislike Poll Feedback
SUB: CHALLENGE over SEND: MOOD voted

IHRE LEHRVERANSTALTUNG

In dieser Lehrveranstaltung lernen Sie alle in der industriellen Praxis relevanten Konzepte und Methoden des **Software Engineering** und dessen "Königsdizziplin" **Software Architecture** kennen – aus der spezifischen Perspektive der Software-Industrie. Die Lehrveranstaltung besteht aus einer Vorlesung mit 10 Modulen und jeweils 3 Stunden, komprimiert auf nur 5 Wochen in 2022/Q4 des WS22/23. Zusätzlich werden Sprechstunden angeboten.

IHRE TEILNAHME

Die Lehrveranstaltung findet, bis auf die Abschlussklausur, ausschließlich online statt. Dabei kommen zwei unterschiedliche Arten der Kommunikation zum Einsatz, welche jede bewußt einen anderen Aspekt der Lehrveranstaltung optimiert.

Einerseits kommen für die **Vorlesung** hochauflösende 1080p30 Video-Streams für die Kombination von Dozent, Inhalten und einem Head-Up-



- **WHAT:**
produce lecture in studio environment, with real-time video mixing. Blend content on tablet, lecturer at table, *Head-Up-Display (HUD)* and special effects. Use unified colouring theme.
- **WHY (DIDACTICS):**
acceptance through consistency and seamless blending.
- **WHY (OTHER):**
Generation-Z expectations (video-streaming style), overall user-experience.

Didactics Crux!

Software Classes

Custom Software Development (CSD)
Commercial development of non-standardized, fully individualized and non-reusable company specific software for a single customer.

Standard Software Development (STD)
Commercial development of standardized, partially customizable and fully reusable domain specific software for a class of customers.

Open Source Software Development (OSS)
Non-commercial development of standardized, highly customizable, and fully reusable generic software for a class of customers.

Class: Business & Data
Software for productivity in the desktop-based office environment.

Class: Machinery & Network
Software for controlling a physical machinery or technical system.

Class: Development & Tools
Software libraries and frameworks of reusable functionality for developing software.

What kind of software development is reasonable for establishing a commercial business, based a reusable product?

1: Custom Software Development 2

2: Standard Software Development 14/18 (78%)

3: Open Source Software 1

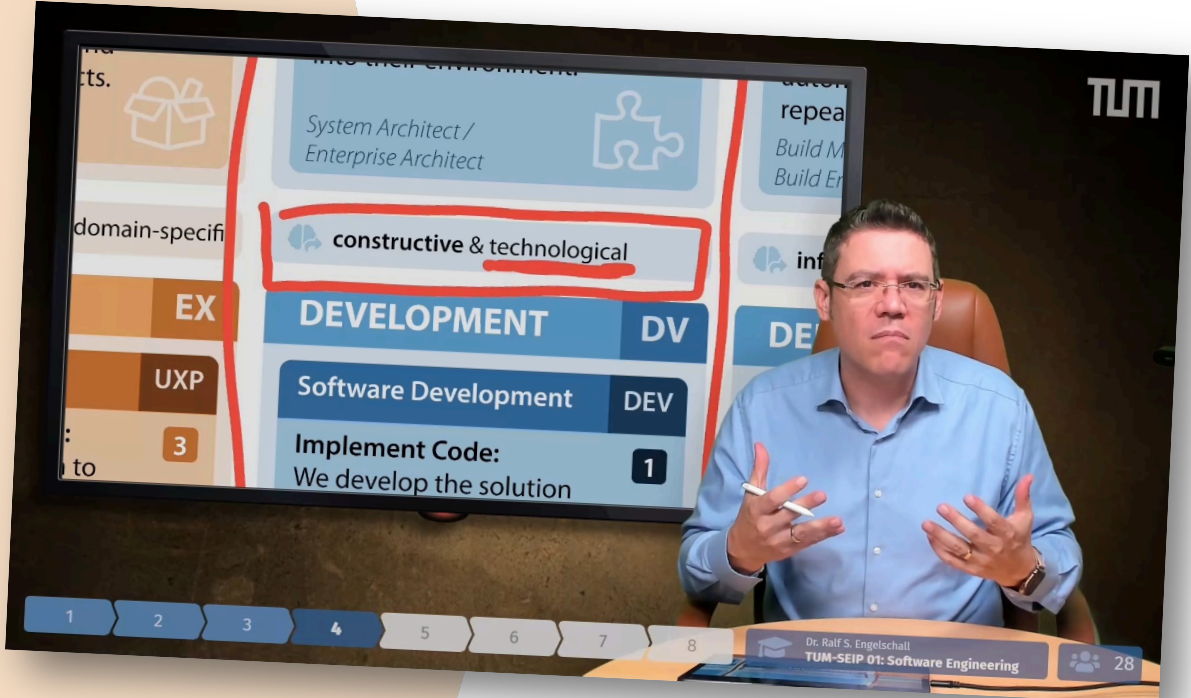
Dr. Ralf S. Engelschall
TUM-SEIP 01: Software Engineering

25



- **WHAT:**
stay at and focus on single diagram for at least 15-20 minutes in total. Zoom around in diagram and on-the-fly annotate it.
- **WHY (DIDACTICS):**
strong content focus during lecture, easily adjust “depth of lecturing”.
- **WHY (OTHER):**
Anti-Death-By-PowerPoint.

Didactics Crux!



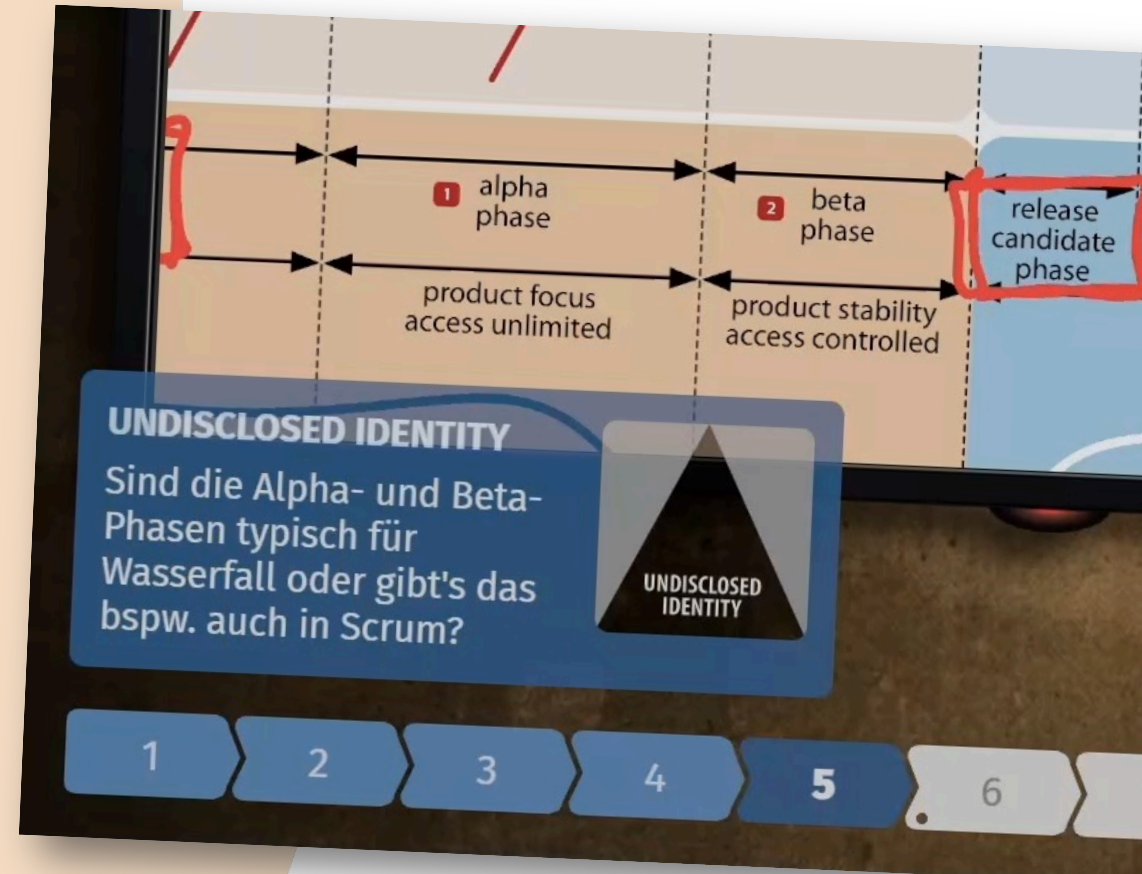
The image shows a man in a blue shirt sitting in front of a large screen displaying a presentation slide. The slide content includes:

- System Architect / Enterprise Architect
- constructive & technological (highlighted with a red box)
- DEVELOPMENT
- Software Development
- Implement Code: We develop the solution

The screen also features a navigation bar at the bottom with numbers 1 through 8, and a footer with the text: Dr. Ralf S. Engelschall, TUM-SEIP 01: Software Engineering, and the number 28. The TUM logo is visible in the top right corner of the screen.



- **WHAT:**
allow students to (optionally anonymously) raise questions and statements at any time, work them off sequentially.
- **WHY (DIDACTICS):**
direct student involvement, no direct lecturer interruption, stronger questions & answers focus, less misunderstandings.
- **WHY (OTHER):**
YouTube/Twitch-style chat communication of *Generation-Z*.





- **WHAT:**
raise (potentially “beyond one’s own nose”) questions about every 10-15 minutes, let students vote anonymously and in real-time.
- **WHY (DIDACTICS):**
continuous student involvement, loosening up lecture, competition fun.
- **WHY (OTHER):**
(none)

Didactics Crux!

What primarily characterizes the ALPHA phase during product development?

1: completing feature set	5
2: gathering feature feedback	9/22 (41%)
3: stabilizing functionality	5
4: final release polishing	0
5: bugfixing	3

Dr. Ralf S. Engelschall
TUM-SEIP 04: SCM & DevOps
25

Gamification Quizzes



Building Block 9/12

What primarily characterizes the ALPHA phase during product development?

- 1: completing feature set
- 2: gathering feature feedback
- 3: stabilizing functionality
- 4: final release polishing
- 5: bugfixing

(total)

Please choose with a numeric vote now by selecting one of the choices 1, 2, ...

1 2 3 4 5 6 7 8 9 10

Dr. Ralf S. Engelschall
TUM-SEIP 04: SCM & DevOps

What primarily characterizes the ALPHA phase during product development?

- 1: completing feature set
- 2: gathering feature feedback
- 3: stabilizing functionality
- 4: final release polishing
- 5: bugfixing

(total) 16

Please choose with a numeric vote now by selecting one of the choices 1, 2, ...

1 2 3 4 5 6 7 8 9 10

Dr. Ralf S. Engelschall
TUM-SEIP 04: SCM & DevOps

What primarily characterizes the ALPHA phase during product development?

- 1: completing feature set 5
- 2: gathering feature feedback 9/22 (41%)
- 3: stabilizing functionality 5
- 4: final release polishing
- 5: bugfixing 3

1 2 3 4 5 6 7 8 9 10

Dr. Ralf S. Engelschall
TUM-SEIP 04: SCM & DevOps

What primarily characterizes the ALPHA phase during product development?

- 1: completing feature set 5
- 2: gathering feature feedback 9/22 (41%)
- 3: stabilizing functionality 5
- 4: final release polishing
- 5: bugfixing 3

1 2 3 4 5 6 7 8 9 10

Dr. Ralf S. Engelschall
TUM-SEIP 04: SCM & DevOps

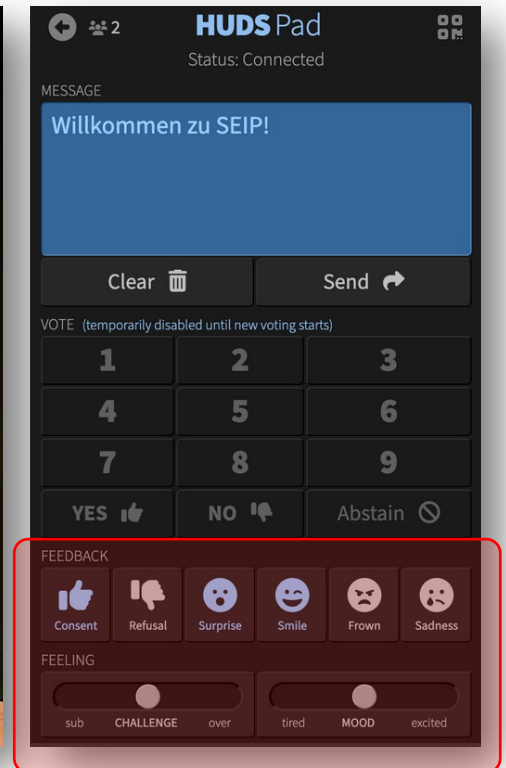
Live Attendee Feedback



Building Block 10/12



- **WHAT:**
allow students to continuously share their emotions (via *Emojis*), content-wise challenge and personal mood.
- **WHY (DIDACTICS):**
real-time student feedback, lecturer speed/depth adjustment, inter-student sympathy.
- **WHY (OTHER):**
annotation of lecture, perception of even “remote controlling” the lecture.



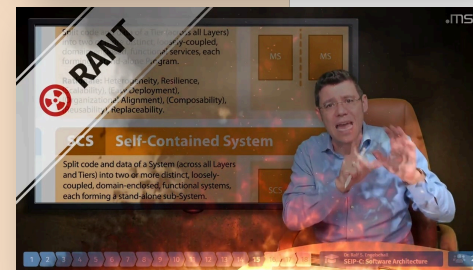
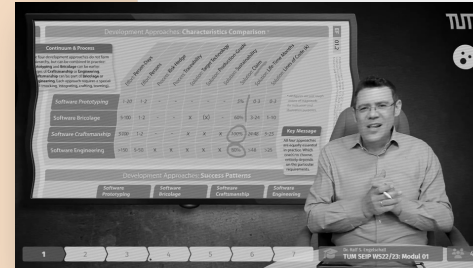
Perception Amplifier



Building Block 11/12



- **WHAT:**
use of curtains, banners, pause clock, and a set of overlaid special effects.
- **WHY (DIDACTICS):**
draw extra attention, amplify content perception.
- **WHY (OTHER):**
(none)



Backstage Pass



Building Block 12/12



- **WHAT:**
during breaks, lecturer gives students live backstage view and explanation of lecture production.
- **WHY (DIDACTICS):**
inspiration and motivation for own presentations.
- **WHY (OTHER):**
proof of media competence, also in *Generation-X*.



My Education Status Quo

Some proving excerpts (in german)
from official lecture evaluations:

[...]die Balance aus informativ und unterhaltsam, eigentlich die gesamte Vorlesung ist überragend.

– TUM, SEIP, WS 22/23

Die Professionalität [...] übersteigt alles was ich bisher in meiner Studentenlaufbahn erleben durfte.

– TUM, SEIP, WS 22/23

Definitiv die beste Vorlesung, die ich je besucht habe!

– TUM, SEIP, WS 22/23

Eine tolles Maß an “Gamification”

– TUM, SEIP, WS 20/21

[...] eine der didaktisch besten Lehrveranstaltungen, die ich JEMALS besucht habe.

– TUM, SEIP, WS 20/21

Die Vorlesung ist bei weitem die aufwändigste und beste, die ich je an der TUM [...] gesehen habe.

– TUM, SEIP, WS 22/23

Die Medien und das Setup sind unfassbar gut!

– TUM, SEIP, WS 21/22





Company: SEA Software Engineering Academy gGmbH
Register: HRB 26 92 37 (Amtsgericht München)
Shareholder: Dr. Ernst Denert, Dr. Ralf S. Engelschall
Director: Dr. Ralf S. Engelschall
VAT-Id (USt-IdNr): DE 351 992 667
Web: <https://softeng.academy>
Email: contact@softeng.academy
Postal: Weblinger Weg 28, 85221 Dachau, DE
IBAN: DE93 700 700 24 0 798 1 798 00
Phone: +49 8131 7799780



**Software
Engineering
Academy**

*Thanks for
your attention!*