Vorlesung Fortgeschrittene Softwaretechnik

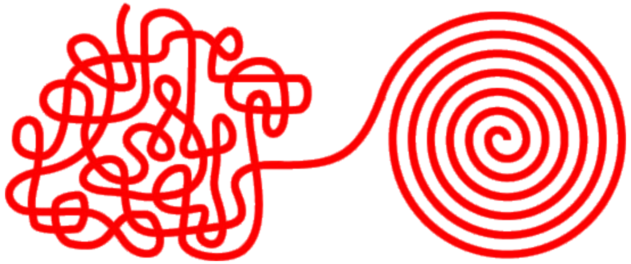
Wintersemester 2024/25

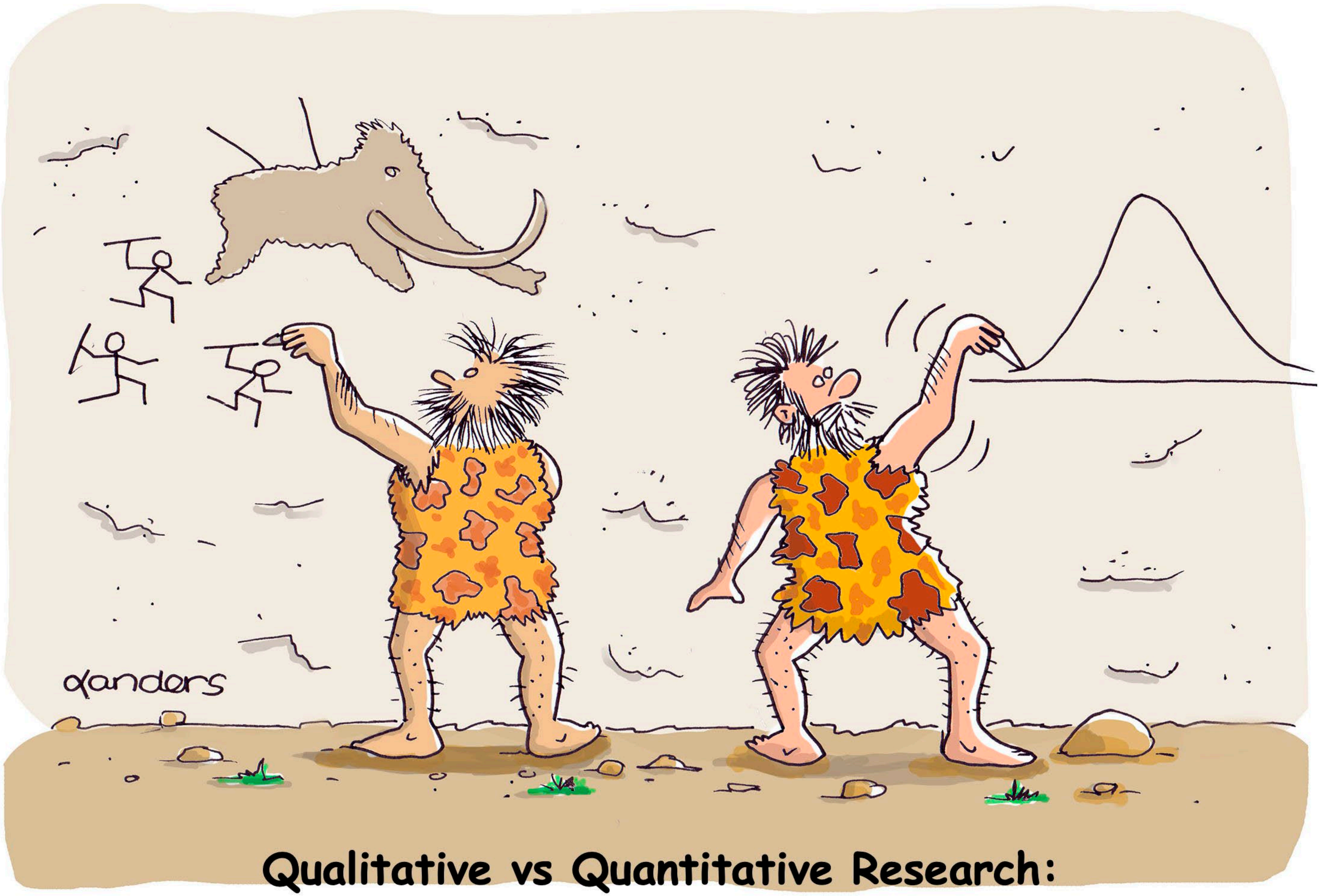
Prof. Dr. Stephan Diehl

Informatik

Universität Trier

Qualitative Data Analysis





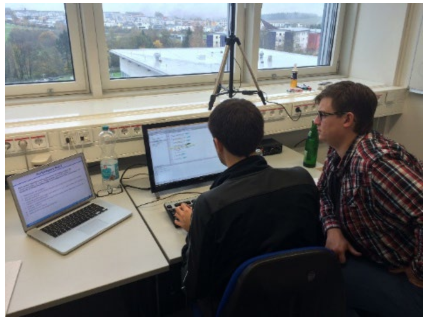
Qualitative vs Quantitative Research:
Tell a Story or Use a Graph?

Qualitative vs. Quantitative Data

Qualitative Data:

- *“Quality”*
- Unstructured information
  (descriptions of behavior/opinions/attitudes as text/image/video)
- Often manual analysis
- *Instruments:* Interviews, open-ended surveys, focus groups, observational studies

“The integration into the code view provides additional context for the profiling visualization.”



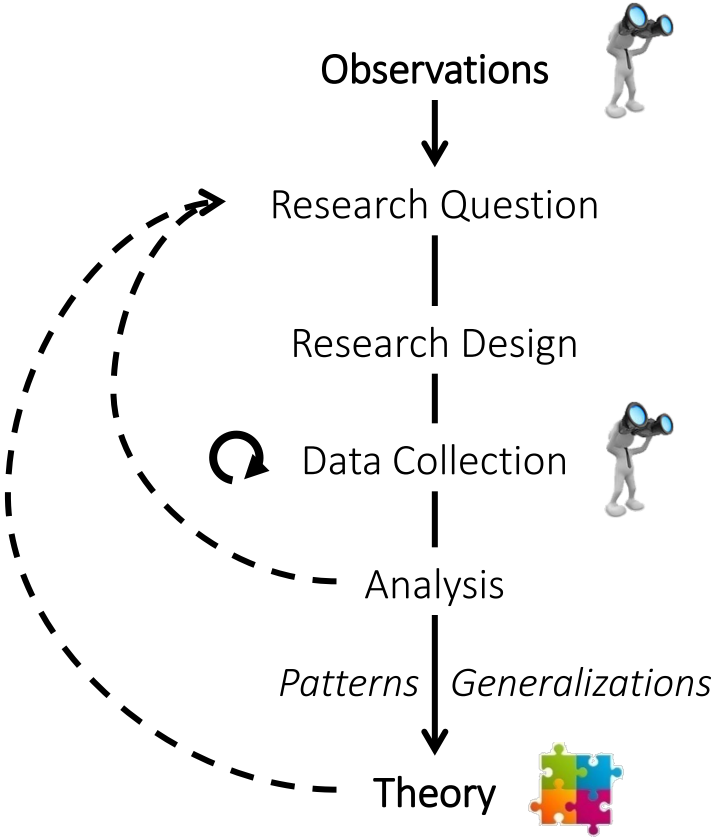
Quantitative Data:

- *“Quantity”*
- Structured information
  (quantification of properties as numerical data)
- Analysis using statistical methods
- *Instruments:* Experiments, closed-ended surveys

Participant 1:
Work experience: 5 years
Time task 1: 23 min.

Participant 2:
Work experience: 4 years
Time task 1: 25 min.

Inductive Research



- **Exploring** new phenomena
- Open-ended
- Process-oriented
- Focus on **qualitative data** (but quantitative data may also be used)
- Generating new theory from data (*grounded theory*)
- **Reiterating** until saturation (*constant comparison*)

Example: *How do software developers interact in a pair-programming setting?*

The Qualitative-Quantitative Debate

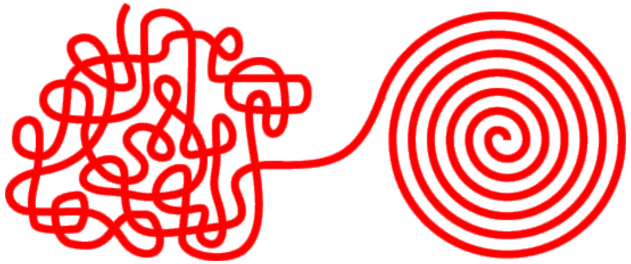
Qualitative Researcher:

- *„The best way to understand phenomenon is viewing it in its context; controlled experiments cannot capture the whole phenomenon.“* **(preference of field studies)**
- *„Researchers are allowed to become immersed in the whole research setting.“* **(adoption of ethnographic methods)**
- *„Research questions may emerge and change during the course of research.“* **(inductive, exploratory research)**
- *„Knowledge of the world is always a human and social construction.“* **(constructivism)**

Quantitative Researcher:

- *“Measure what is measurable, and make measurable what is not so.”* *(attributed to Galilei)* **(quantification)**
- *“The role of the researcher is limited to data collection and interpretation through objective (statistical) methods.”* **(positivism, statistical methods)**
- *”Concepts must be operationalized and hypotheses must be deduced from theory.”* **(deductive research)**
- *“Explanations must demonstrate causality.”* **(controlled experiments)**

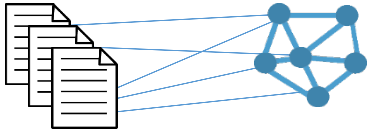
Qualitative Data Analysis



Grounded Theory

Grounded theory is a systematic methodology that aims at **constructing theory through the analysis of data.**

- Developed around 1965 by Barney Glaser and Anselm Strauss
- Begins with a question (or just with data collection)
- The researcher analyzes the data and tags relevant parts with **codes**
- Analysis and data collection are **iterative processes**, codes are constantly revised (constant comparison)
- **Memos*** are used to structure and document the process
- Collecting data until **saturation** is achieved
- Codes are then **grouped** into concepts
- Concepts are grouped into categories
- Categories can become foundation of new theory



Theory is "grounded"
in data

*Memos record hypotheses and thoughts about the data and possible connections.

From Code to Theory

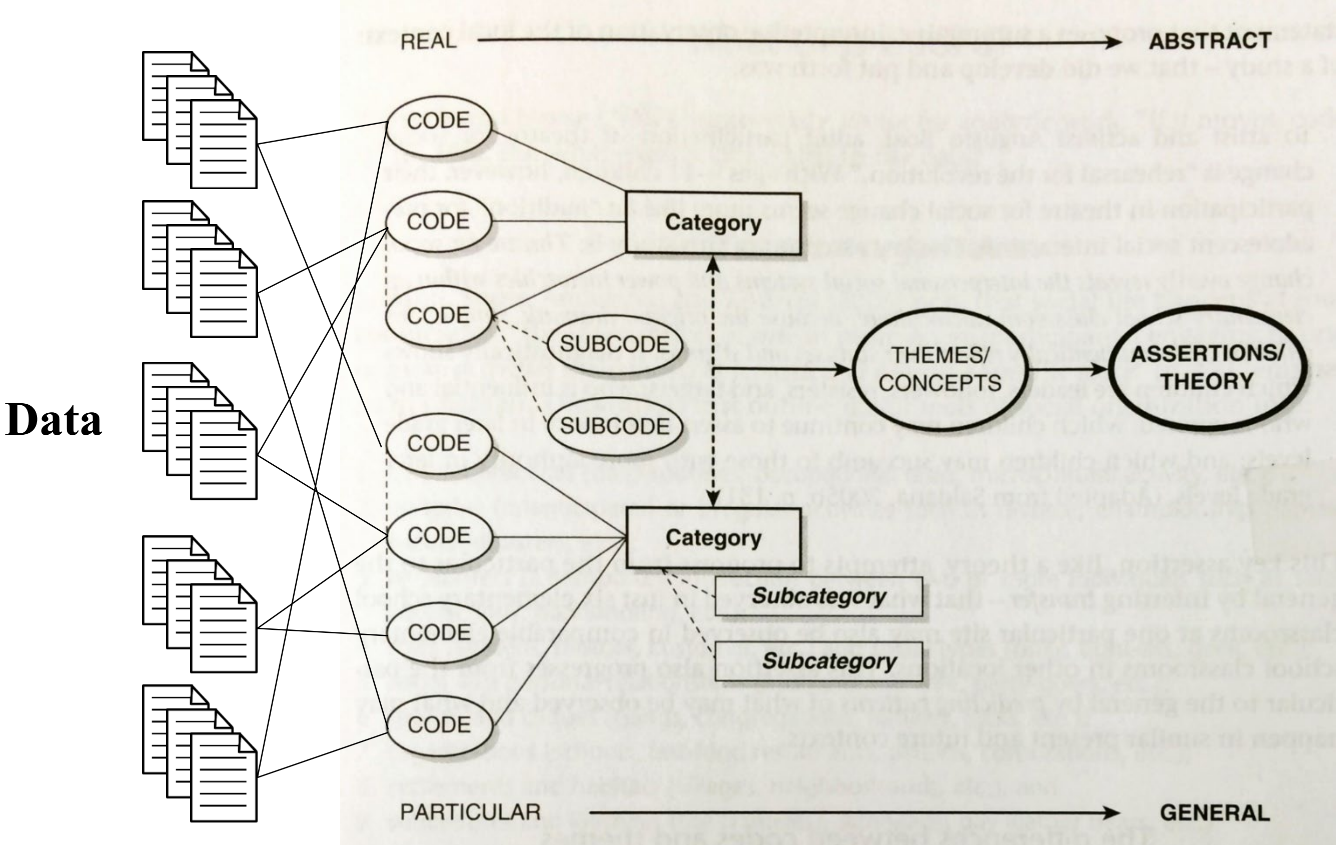
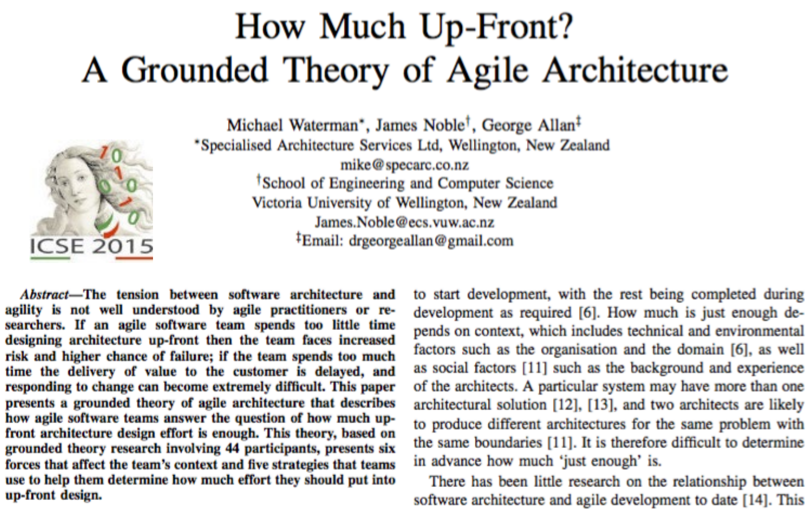


Figure 1.1 A streamlined codes-to-theory model for qualitative inquiry

[Saldana13]

Example



How Much Up-Front?
A Grounded Theory of Agile Architecture

Michael Waterman[*], James Noble[†], George Allan[‡]
[*]Specialised Architecture Services Ltd, Wellington, New Zealand
mike@specarc.co.nz
[†]School of Engineering and Computer Science
Victoria University of Wellington, New Zealand
James.Noble@ecs.vuw.ac.nz
[‡]Email: drgeorgeallan@gmail.com

ICSE 2015

***Abstract*—The tension between software architecture and agility is not well understood by agile practitioners or researchers. If an agile software team spends too little time designing architecture up-front then the team faces increased risk and higher chance of failure; if the team spends too much time the delivery of value to the customer is delayed, and responding to change can become extremely difficult. This paper presents a grounded theory of agile architecture that describes how agile software teams answer the question of how much up-front architecture design effort is enough. This theory, based on grounded theory research involving 44 participants, presents six forces that affect the team's context and five strategies that teams use to help them determine how much effort they should put into up-front design.**

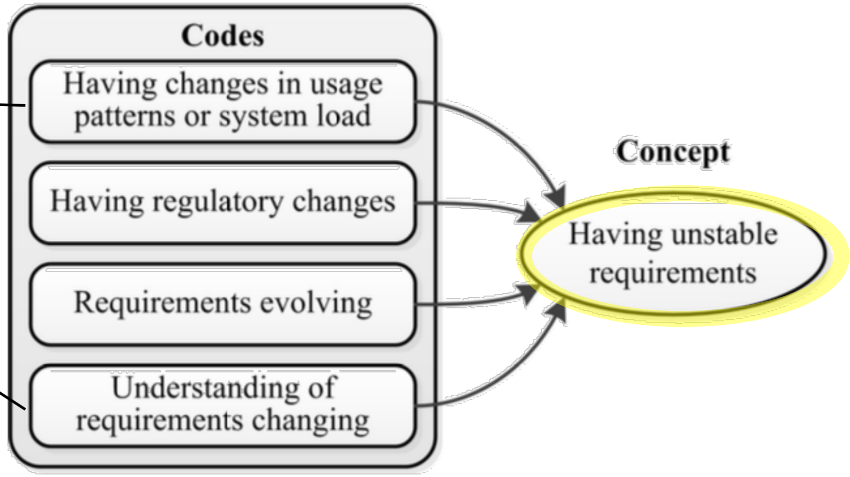
to start development, with the rest being completed during development as required [6]. How much is just enough depends on context, which includes technical and environmental factors such as the organisation and the domain [6], as well as social factors [11] such as the background and experience of the architects. A particular system may have more than one architectural solution [12], [13], and two architects are likely to produce different architectures for the same problem with the same boundaries [11]. It is therefore difficult to determine in advance how much 'just enough' is.

There has been little research on the relationship between software architecture and agile development to date [14]. This

- Interviewed 44 participants
- Result: Grounded theory involving 11 categories (”forces and strategies”)

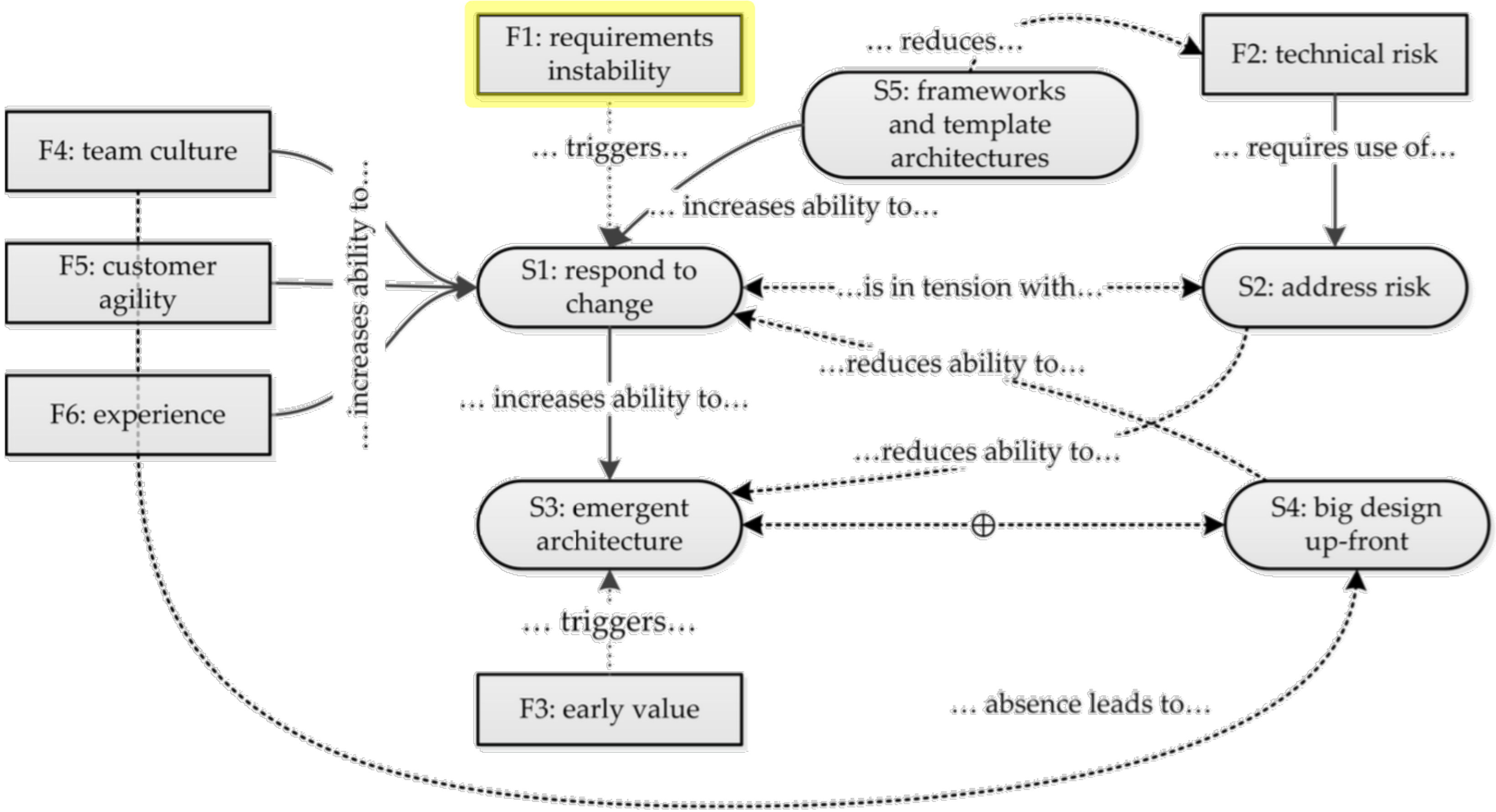
“[If you put too much detail into your requirements] there’s a fat chance that by the time you get around to starting the work your world has changed.” (P3, development manager)

“I don’t know if the actual requirements ever changed but our understanding of them changed enormously.” (P13, architect)


Codes
Having changes in usage patterns or system load
Having regulatory changes
Requirements evolving
Understanding of requirements changing
Concept
Having unstable requirements

Here: categories partitoned into forces and strategies.

Example



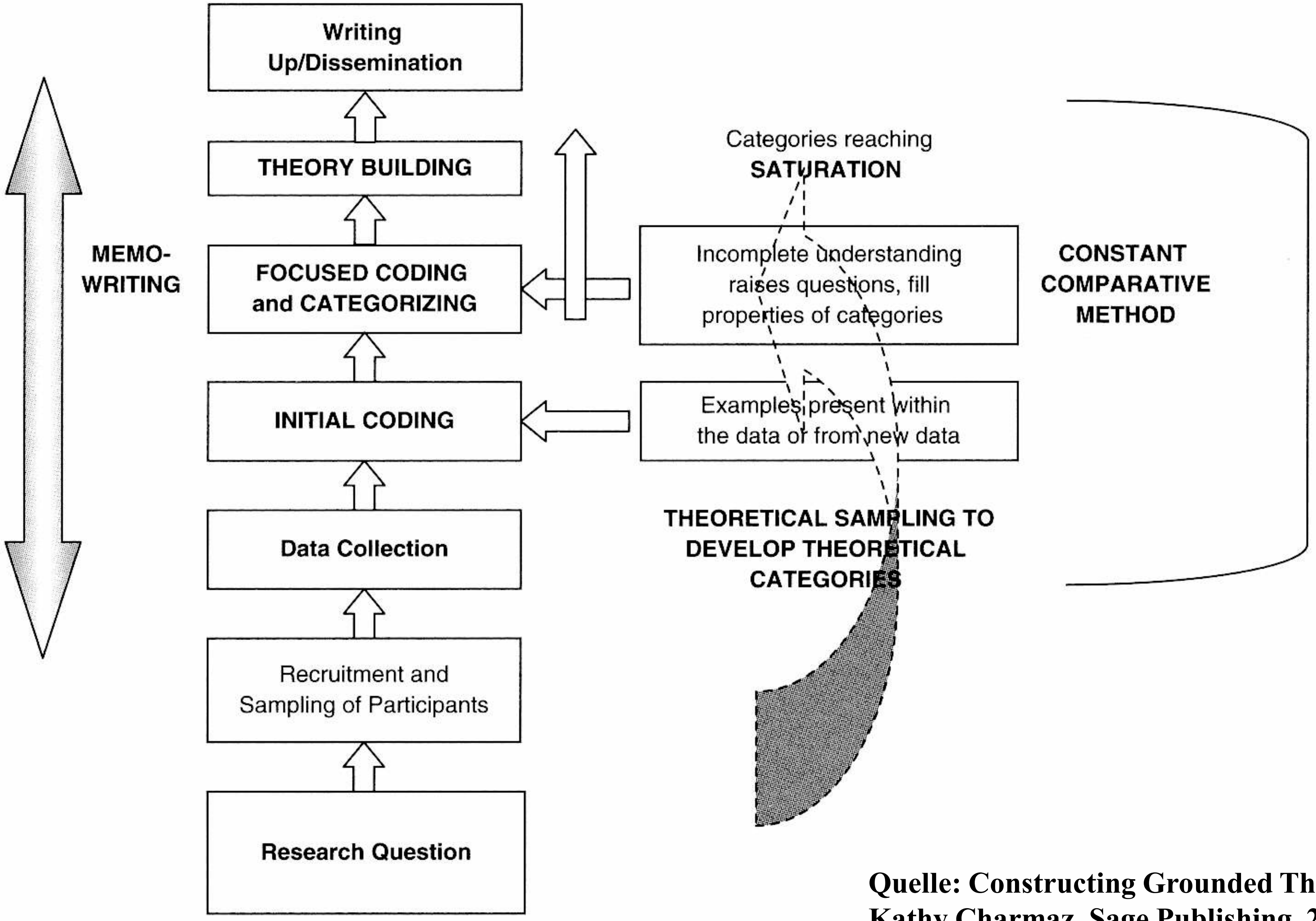
Waterman et al. – How Much Up-Front? A Grounded Theory of Agile Architecture (ICSE 2015)

Coding

- [Charmaz14]: **Two stages**: Initial (open) coding and focused coding (includes categorization)
- [Saldana13]: **First cycle** (open) vs. **second cycle** (focused) coding methods (hybrids possible)
- Avoid mere description ("information gathering" not "reading the schedule")
- Iterative process, **constant comparison**
- Can be useful even if the process does not result in theory
- Coding can be seen as **quantification of qualitative data** (descriptive statistics possible)
- Coding depends on researcher (accepting vs. trying to achieve inter-rater agreement)

Quellen:
Johnny Saldana – The Coding Manual for Qualitative Researchers (2nd edition, 2013, Sage)
Juliet Corbin and Anselm Strauss – Basics of Qualitative Research (3rd edition, 2008, Sage)
Kathy Charmaz – Constructing Grounded Theory (2nd edition, 2014, Sage)

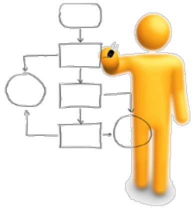
Charmaz's Approach


Writing Up/Dissemination
THEORY BUILDING
FOCUSED CODING and CATEGORIZING
INITIAL CODING
Data Collection
Recruitment and Sampling of Participants
Research Question
MEMO-WRITING
Categories reaching SATURATION
Incomplete understanding raises questions, fill properties of categories
Examples present within the data of from new data
THEORETICAL SAMPLING TO DEVELOP THEORETICAL CATEGORIES
CONSTANT COMPARATIVE METHOD

Quelle: Constructing Grounded Theory, Kathy Charmaz, Sage Publishing, 2014

First Cycle Coding Methods

- **Structural Coding:** Structure data by assigning content-based phrases to large segments that relate to a specific research question.



- **Open/Initial Coding:** Open search for and coding of statements related to research question. Unit of analysis: lines, sentences, paragraphs.

Initial Coding

Focus on certain aspects while doing open coding:

Data-driven codes

- **In Vivo Coding:** Using words or short phrases from data records as codes.
- **Process Coding:** Using only gerunds to connote action in the data.
- **Emotion Coding:** Labeling the emotions recalled and/or experienced by the participant.
- **Values Coding:** Coding data that reflects a participant’s values, attitudes, and beliefs.
- **Versus Coding:** Coding (in binary terms) conflicts between people, processes, concepts, etc.

Theory-driven codes

- **Hypothesis Coding:** Applying a predetermined list of codes to assess a hypothesis. Both are developed by the researcher from theory before beginning the analysis.
- **Protocol Coding:** Following a research protocol (detailed procedural guidelines for data collection and analysis). Usually, a comprehensive list of codes and categories is provided.

Researchers have proposed many other „codings“ (see f.e. https://sites.google.com/site/qualitativecodebook/) including motif, topic, concept, holistic, pattern, theoretical, eleborative, longitudinal coding.

Focused Coding

- **Revision** of codes (e.g. more precise wording), **merging** of codes, **dropping** of infrequent or unimportant codes
- Focus on most salient (e.g. most frequent, most significant) codes
- With each cycle, number of **codes** should decrease
- **Search for patterns**
- **Connect and structure** codes using categories
- Describe relationships in **memos** or using an annotated graph
- **Goal:** Come up with one central/core category (not always necessary or possible)

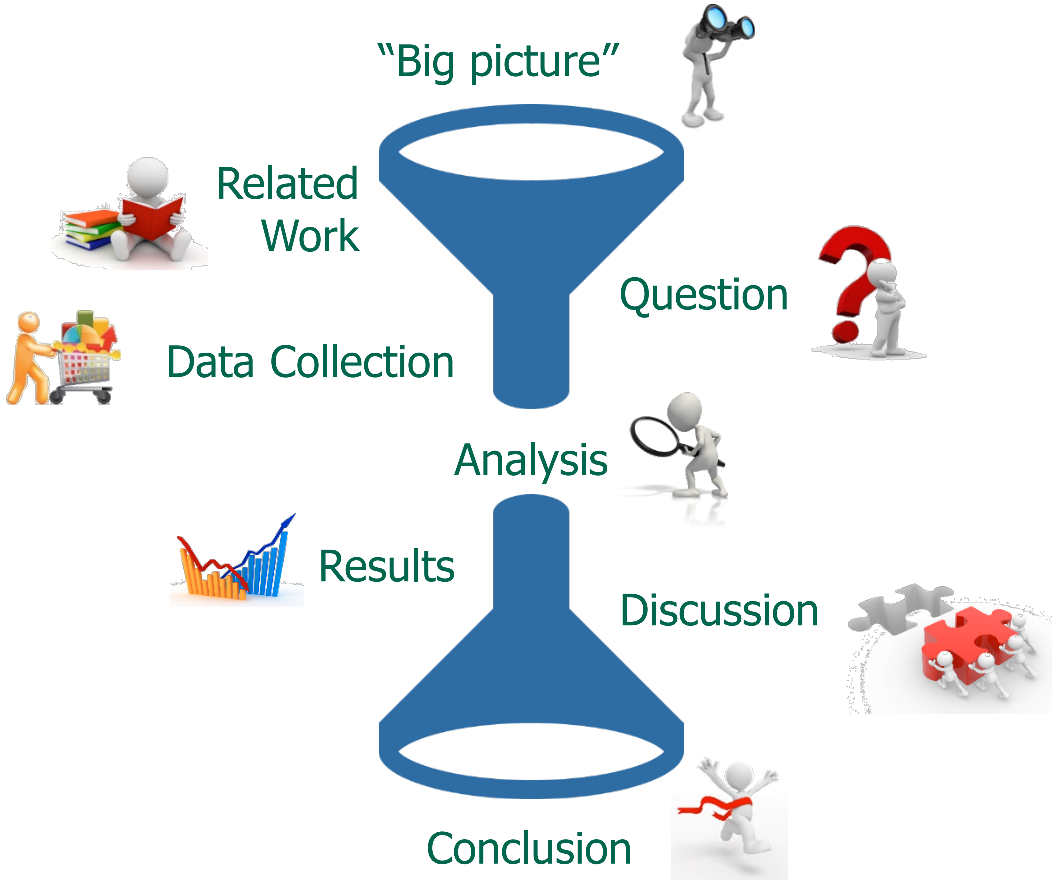
Memos

- Different media feasible for memo writing (notebook, text files, CAQDAS* software, etc.)
- Memos **document** the course of research
- Memos are a tool to **reflect** on coding process, code choices, and emergent patterns
- Memos can contain thoughts, questions, assumptions, definitions, problems, graphs documenting relationships, etc.

*Computer-Assisted/Aided Qualitative Data Analysis Software

Properties of „Good“ Research

- Reproducibility of Research



“Big picture”

Related Work

Question

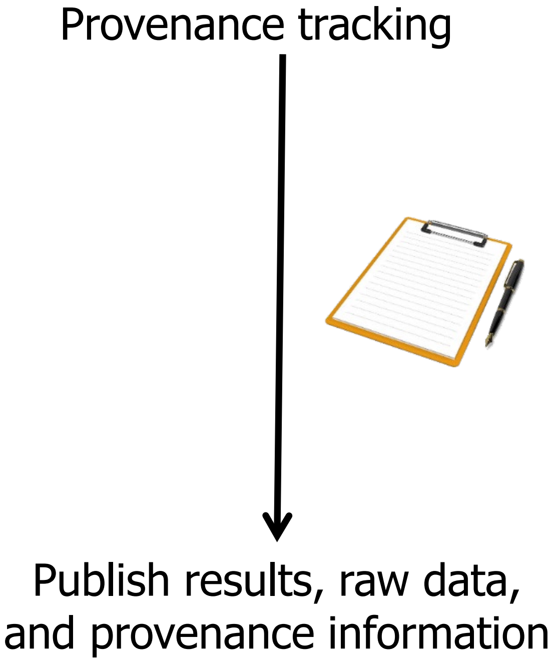
Data Collection

Analysis

Results

Discussion

Conclusion



Provenance tracking

Publish results, raw data, and provenance information

Guidelines for Analyzing Qualitative Data

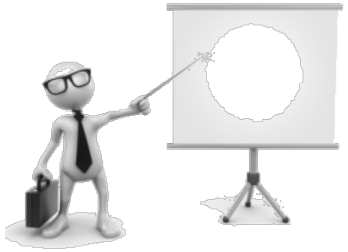
- **Know yourself**, your biases, and preconceptions.
- **Know your question.**
- **Be flexible.**
- **Exhaust the data.** Try to account for all the data in the texts, then publicly acknowledge the unexplained and remember the next principle.
- **Celebrate anomalies.** They are the windows to insight.
- **Get critical feedback.** The solo analyst is a great danger to self and others. Consult others and keep looking for alternative interpretations.
- **Be explicit.** Share the details with yourself, your team members, and your audiences.

Miller&Crabtree – Doing Qualitative Research (1999)

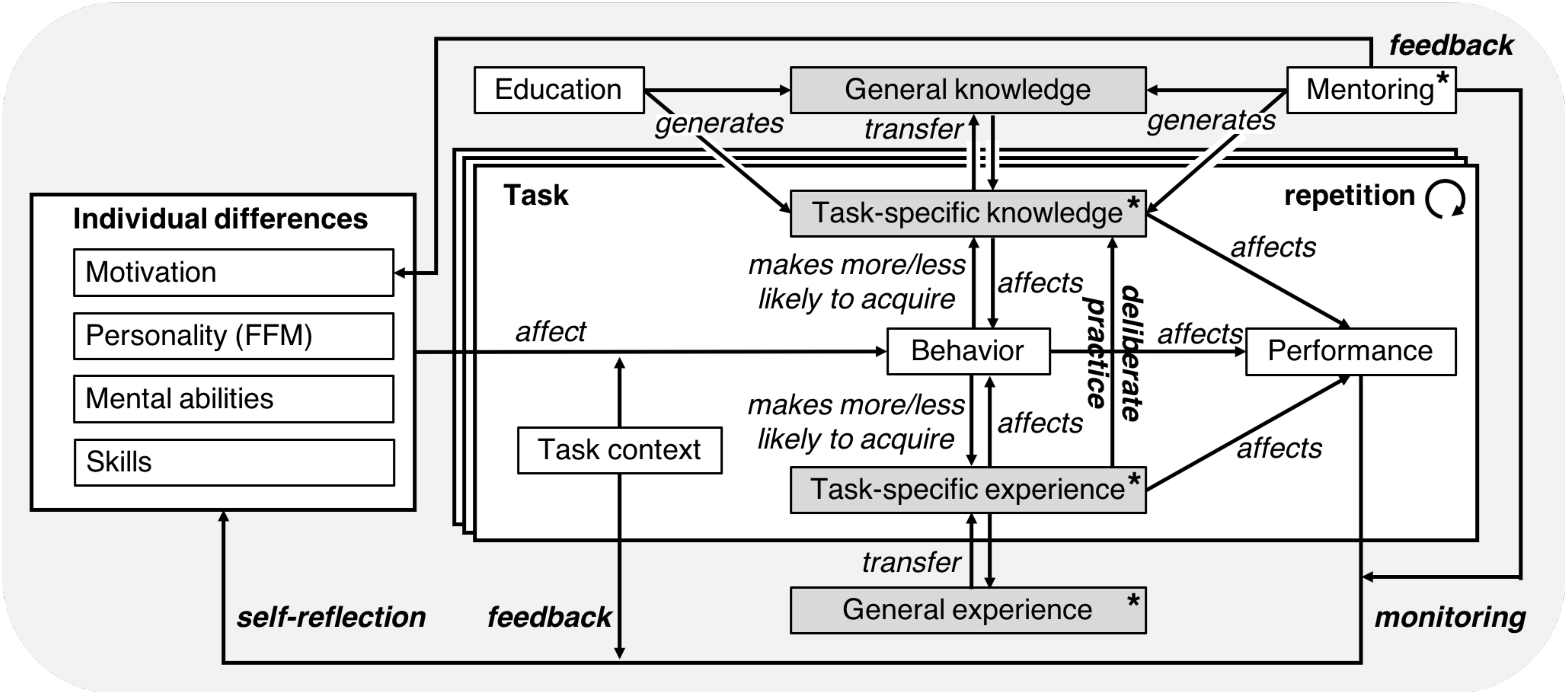
Reporting Qualitative Results

Reporting Qualitative Results

- **Graph** showing concepts and their connections
- Name and describe **concepts and related codes**
- Include **quotes** that are representative for a category/concept
- Describe observed **patterns**
- **Descriptive statistics**
- Publish codes and memos (if confidentiality permits that)
- Interpretations should always be **grounded in data**



Example: Conceptual Theory



Baltes and Diehl – Towards a Theory of Software Development Expertise (FSE 2018)

Example: Descriptive Statistics for Codes

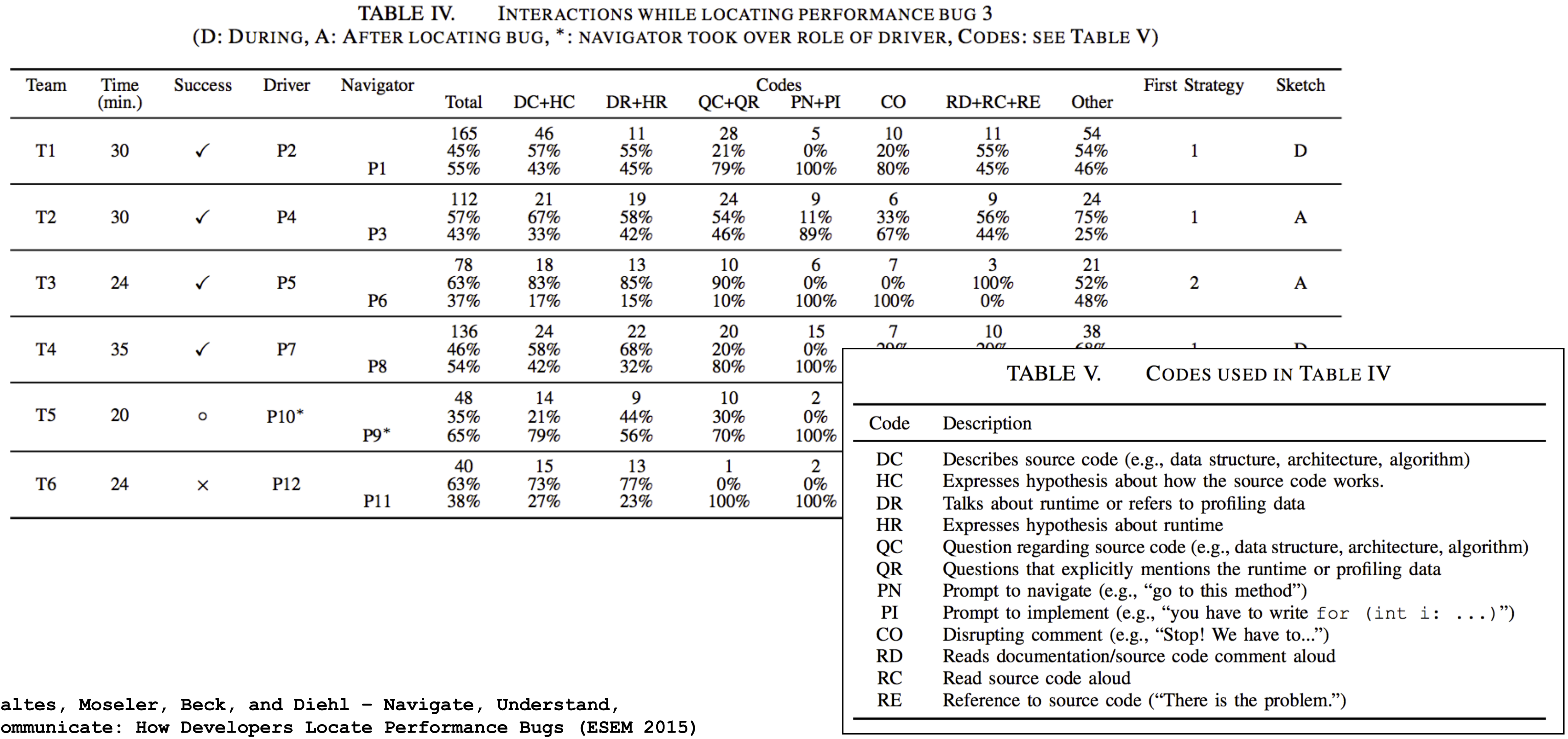


TABLE IV. INTERACTIONS WHILE LOCATING PERFORMANCE BUG 3
(D: DURING, A: AFTER LOCATING BUG, *: NAVIGATOR TOOK OVER ROLE OF DRIVER, CODES: SEE TABLE V)

Team	Time (min.)	Success	Driver	Navigator	Codes								First Strategy	Sketch
					Total	DC+HC	DR+HR	QC+QR	PN+PI	CO	RD+RC+RE	Other		
T1	30	✓	P2	P1	165 45% 55%	46 57% 43%	11 55% 45%	28 21% 79%	5 0% 100%	10 20% 80%	11 55% 45%	54 54% 46%	1	D
T2	30	✓	P4	P3	112 57% 43%	21 67% 33%	19 58% 42%	24 54% 46%	9 11% 89%	6 33% 67%	9 56% 44%	24 75% 25%	1	A
T3	24	✓	P5	P6	78 63% 37%	18 83% 17%	13 85% 15%	10 90% 10%	6 0% 100%	7 0% 100%	3 100% 0%	21 52% 48%	2	A
T4	35	✓	P7	P8	136 46% 54%	24 58% 42%	22 68% 32%	20 20% 80%	15 0% 100%	7 29%	10 20%	38 68%	1	D
T5	20	○	P10*	P9*	48 35% 65%	14 21% 79%	9 44% 56%	10 30% 70%	2 0% 100%					
T6	24	×	P12	P11	40 63% 38%	15 73% 27%	13 77% 23%	1 0% 100%	2 0% 100%					

TABLE V. CODES USED IN TABLE IV

Code	Description
DC	Describes source code (e.g., data structure, architecture, algorithm)
HC	Expresses hypothesis about how the source code works.
DR	Talks about runtime or refers to profiling data
HR	Expresses hypothesis about runtime
QC	Question regarding source code (e.g., data structure, architecture, algorithm)
QR	Questions that explicitly mentions the runtime or profiling data
PN	Prompt to navigate (e.g., "go to this method")
PI	Prompt to implement (e.g., "you have to write `for (int i: ...)`")
CO	Disrupting comment (e.g., "Stop! We have to...")
RD	Reads documentation/source code comment aloud
RC	Read source code aloud
RE	Reference to source code ("There is the problem.")

Baltes, Moseler, Beck, and Diehl – Navigate, Understand, Communicate: How Developers Locate Performance Bugs (ESEM 2015)

Reporting Qualitative Results: Context

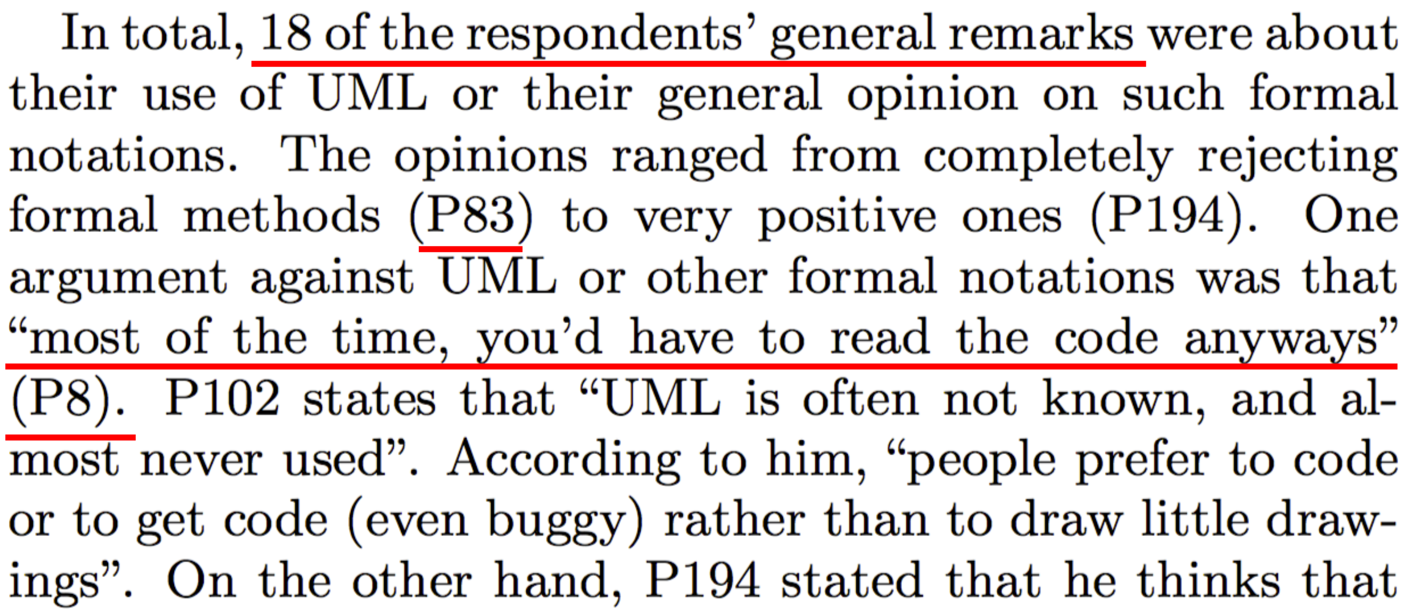
- **Context** is important when reporting qualitative analysis (role of researcher, context of data collection, etc.)

- Reporting **sampling and methodology:**
  - How were subjects/artifacts selected?
    (e.g. convenience or probability sampling)
  - Who were the participants/which artifacts did you select?
    (demographics, metadata)
  - How was the data collected?
    (interviews, questionnaire, manual inspection/collection)
  - How was the analysis conducted?
    (e.g. transcription of data, paper or CAQDAS software, one or multiple researchers, number of iterations)?

Reporting Qualitative Results: Discussion

- Reporting **results and discussion**:
  - Tell a **story** and highlight the most interesting findings (keep research question in mind)
  - Show connection of data and interpretation (**grounding**)
  - Also report **outliers**/cases that do not fit interpretation
  - Use representative **verbatim quotes** for illustration (but don’t overuse them)
  - Care about **confidentiality**/anonymity (quotes, pseudonyms)
  - **Structure**: Report each category/theme in a separate chapter, then discuss findings in relation to existing research OR discuss results immediately when reporting findings

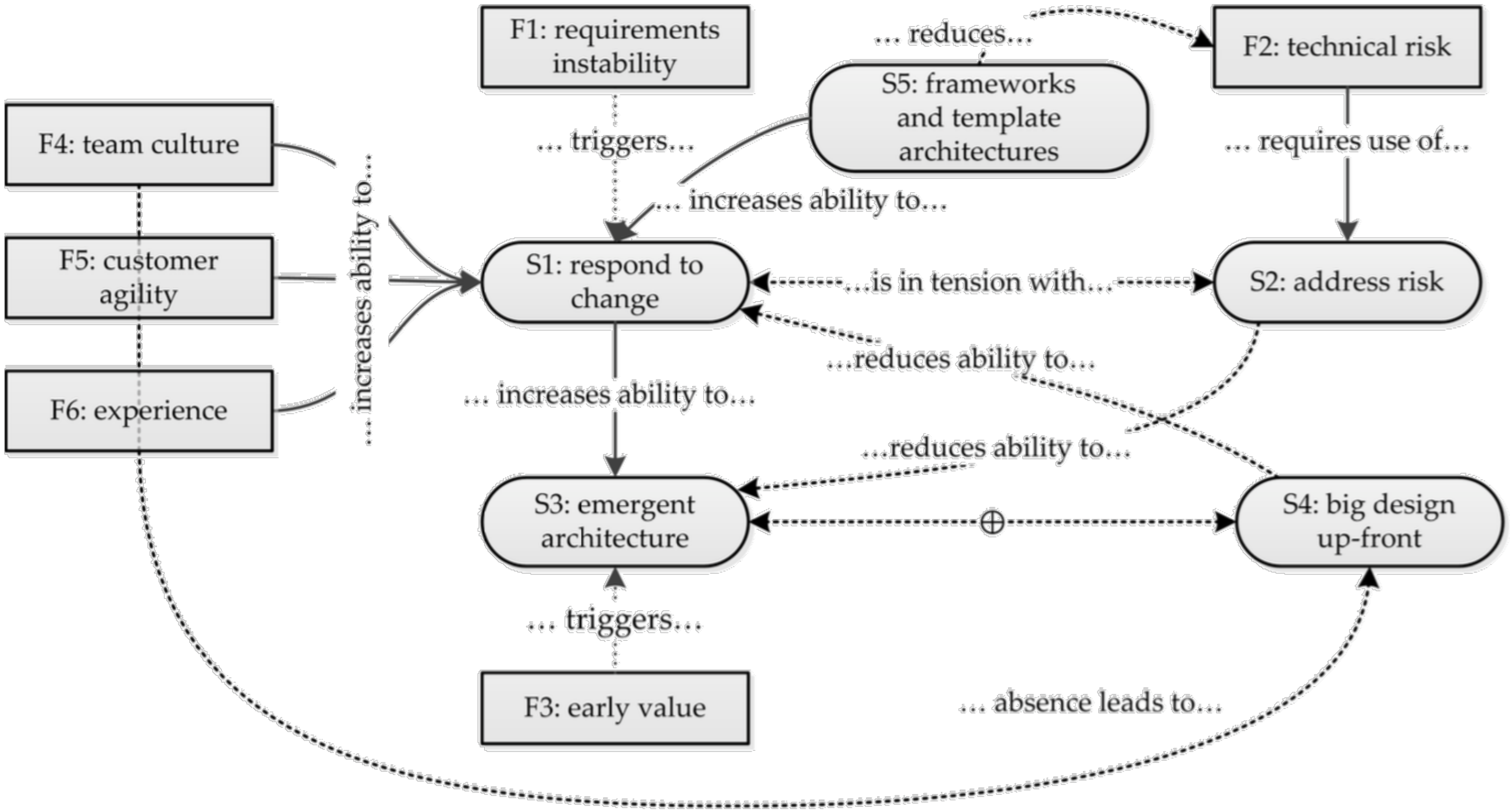
Example: Presenting Qualitative Results



In total, 18 of the respondents' general remarks were about their use of UML or their general opinion on such formal notations. The opinions ranged from completely rejecting formal methods (P83) to very positive ones (P194). One argument against UML or other formal notations was that "most of the time, you'd have to read the code anyways" (P8). P102 states that "UML is often not known, and almost never used". According to him, "people prefer to code or to get code (even buggy) rather than to draw little drawings". On the other hand, P194 stated that he thinks that

Baltes und Diehl – Sketches and Diagrams in Practice (FSE 2014)

Example: Grounded Theory



Waterman et al. – How Much Up-Front? A Grounded Theory of Agile Architecture (ICSE 2015)

Reporting Qualitative Results: Conclusion

- The **conclusion** highlights the main findings and emphasizes what the study adds to existing knowledge

- Some advice:
  - [Chenail95]: “Openness”, “data is the star”, and “juxtaposition”
  - Publish final coding scheme, actual codings, and memos (*reproducible research*)
  - Use **tables** to report information about participants, coding scheme, questionnaire, etc.
  - If you did some quantification (e.g. counting codings), you can also use **bar charts**
  - Present connections between categories in a **graph**
  - Don’t use **percentages or proportions** if you have fewer than 100 answers (25 answers: *“20 participants reported that…”* instead of *“80% of the participants reported that...”*)
  - Discuss results with colleagues

Threats to Validity

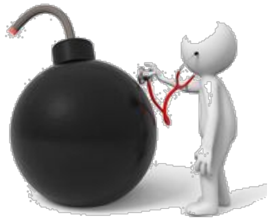
- Address **researcher bias** (always present in qualitative research)
- Mitigation possible:
  - Triangulation
  - Iterating research/constant comparison
  - Try to achieve inter-rater agreement

- Validity criteria:
  - **Credibility:** Are the results credible from the perspective of the participants?
  - **Transferability:** Can the results be generalized or transferred to another context?
  - **Dependability:** Was all data collected in the same setting/context?
  - **Confirmability:** Can other researchers confirm the results?

Threats to Validity
of Quantitative Research

Zum Vergleich

- Validity criteria:
  - **Internal validity:** Can we draw causal conclusions based on the study?
  - **External validity:** Can the results be generalized to the whole population?
  - **Construct Validity:** Were our operationalizations/measurements valid?
  - **Reliability:** Are our measurements consistent (same context, same results)?
  - **Objectivity:** Did the researcher influence the results?



Grounded Requirements Engineering

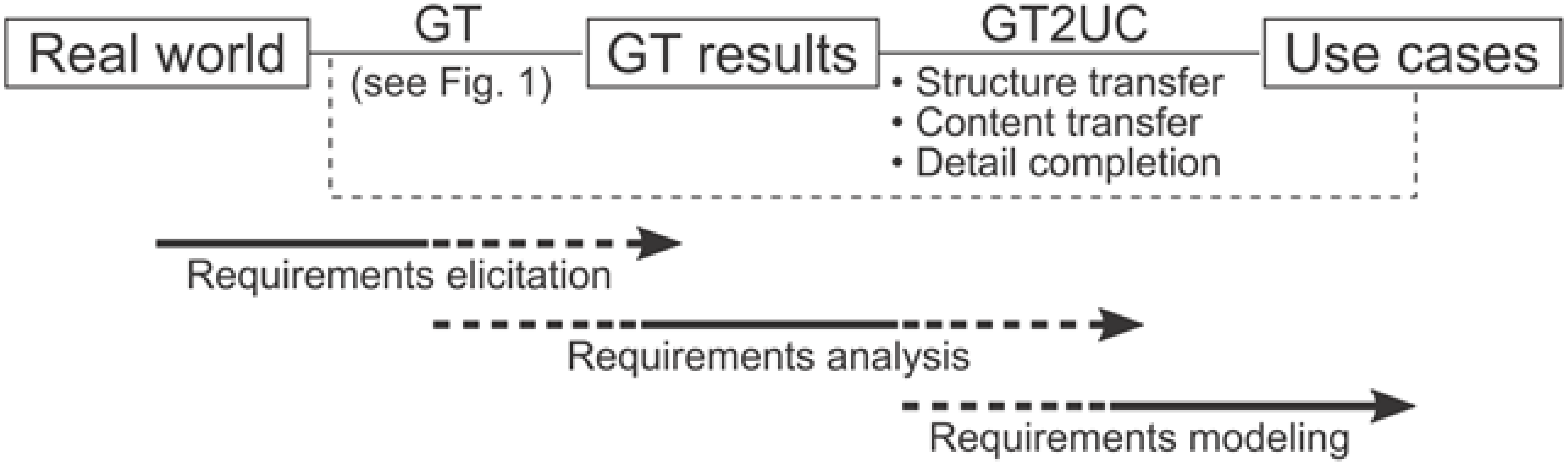
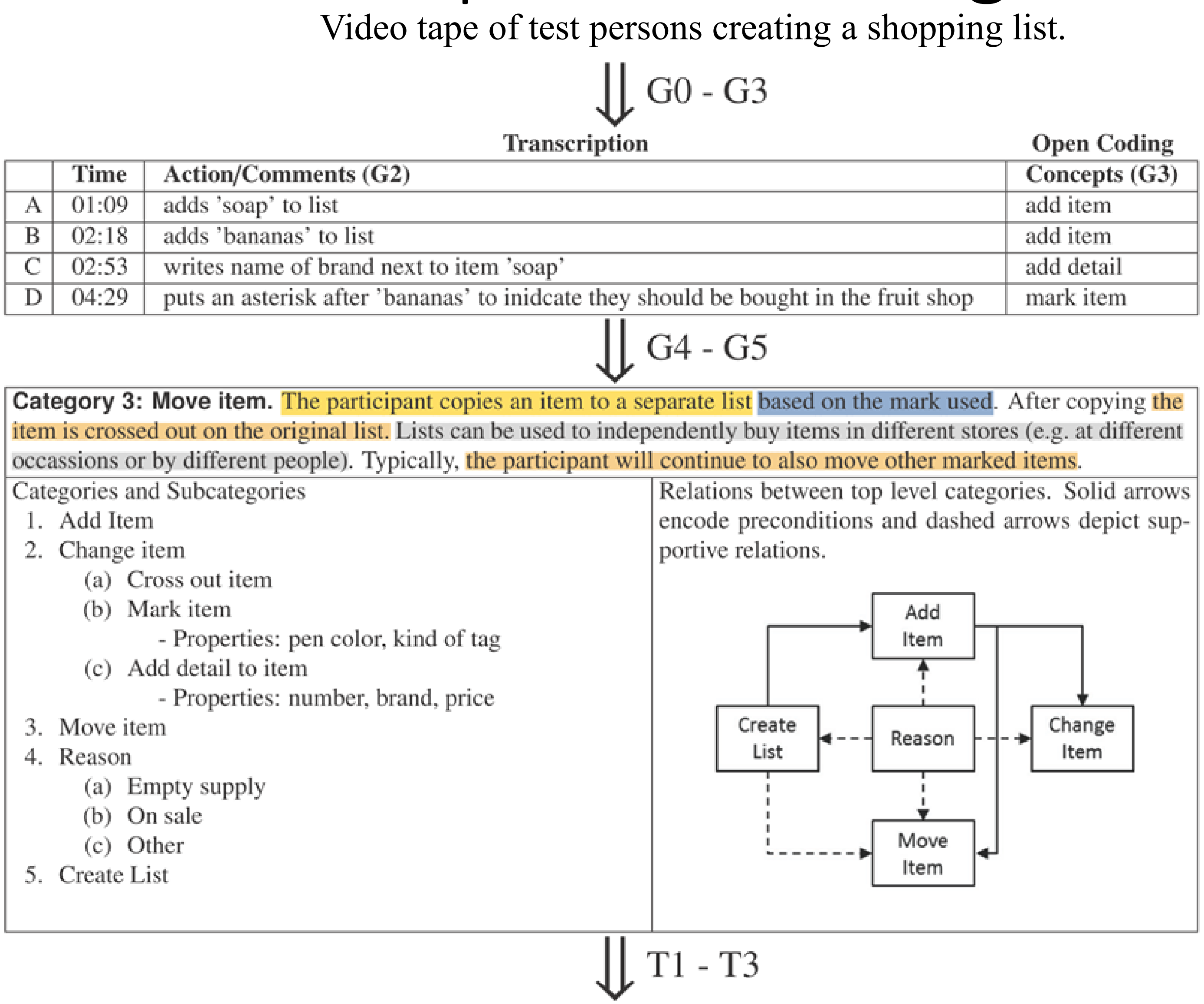


Fig. 2. Grounded Requirements Engineering.

Würfel, Lutz & Diehl – Grounded requirements engineering: An approach to use case driven requirements engineering (The Journal of Systems and Software, 2016)

Grounded Requirements Engineering



Video tape of test persons creating a shopping list.

⇓ G0 - G3

		Transcription	Open Coding
	Time	Action/Comments (G2)	Concepts (G3)
A	01:09	adds 'soap' to list	add item
B	02:18	adds 'bananas' to list	add item
C	02:53	writes name of brand next to item 'soap'	add detail
D	04:29	puts an asterisk after 'bananas' to inidcate they should be bought in the fruit shop	mark item

⇓ G4 - G5

Category 3: Move item. The participant copies an item to a separate list based on the mark used. After copying the item is crossed out on the original list. Lists can be used to independently buy items in different stores (e.g. at different occassions or by different people). Typically, the participant will continue to also move other marked items.

Categories and Subcategories

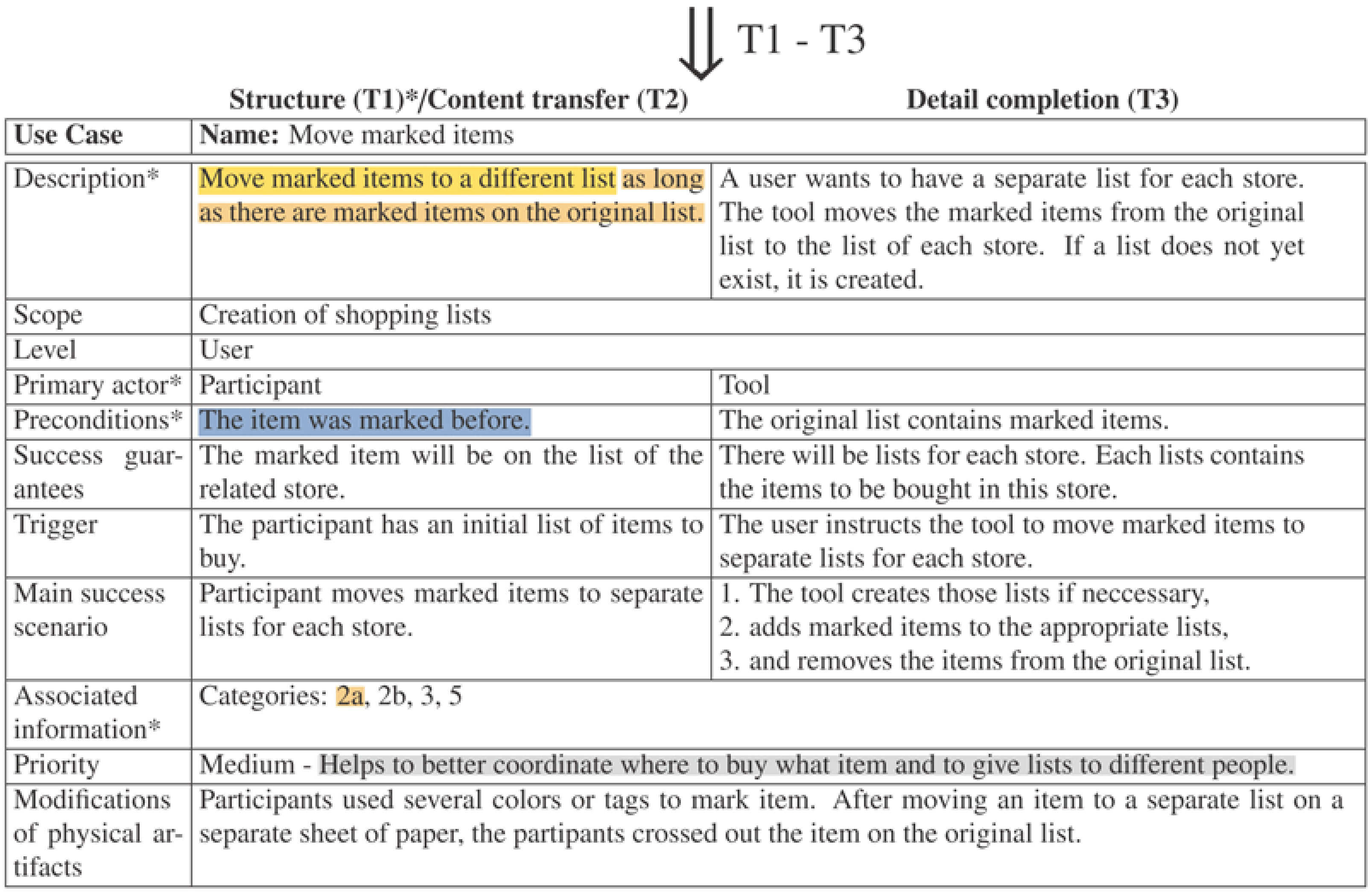
1. Add Item
2. Change item
   (a) Cross out item
   (b) Mark item
       - Properties: pen color, kind of tag
   (c) Add detail to item
       - Properties: number, brand, price
3. Move item
4. Reason
   (a) Empty supply
   (b) On sale
   (c) Other
5. Create List

Relations between top level categories. Solid arrows encode preconditions and dashed arrows depict supportive relations.

⇓ T1 - T3

Würfel, Lutz & Diehl – Grounded requirements engineering: An approach to use case driven requirements engineering (The Journal of Systems and Software, 2016)

Grounded Requirements Engineering



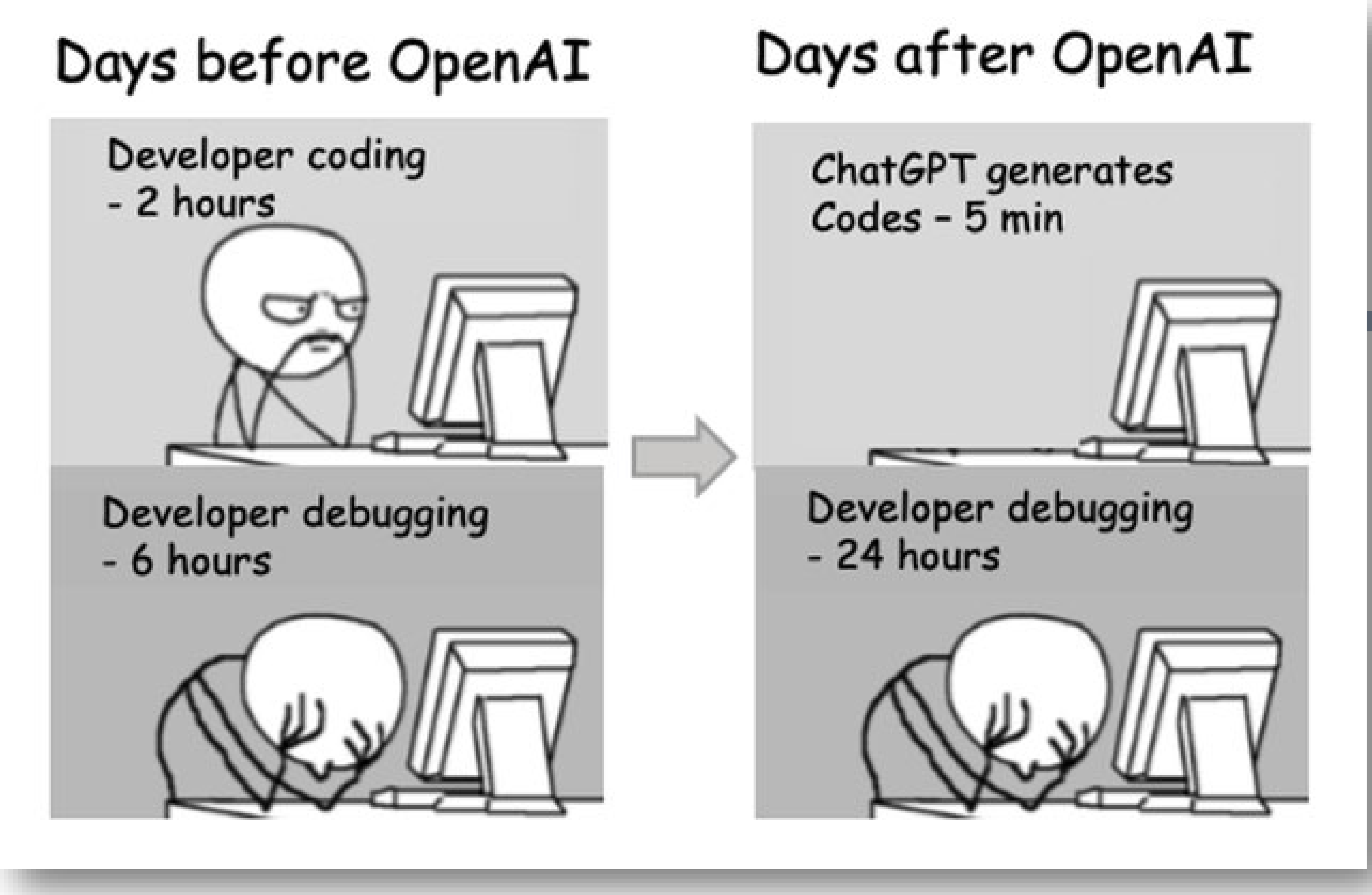
⇓ T1 - T3

	Structure (T1)*/Content transfer (T2)	Detail completion (T3)
Use Case	**Name:** Move marked items	
Description*	Move marked items to a different list as long as there are marked items on the original list.	A user wants to have a separate list for each store. The tool moves the marked items from the original list to the list of each store. If a list does not yet exist, it is created.
Scope	Creation of shopping lists	
Level	User	
Primary actor*	Participant	Tool
Preconditions*	The item was marked before.	The original list contains marked items.
Success guarantees	The marked item will be on the list of the related store.	There will be lists for each store. Each lists contains the items to be bought in this store.
Trigger	The participant has an initial list of items to buy.	The user instructs the tool to move marked items to separate lists for each store.
Main success scenario	Participant moves marked items to separate lists for each store.	1. The tool creates those lists if neccessary, 2. adds marked items to the appropriate lists, 3. and removes the items from the original list.
Associated information*	Categories: 2a, 2b, 3, 5	
Priority	Medium - Helps to better coordinate where to buy what item and to give lists to different people.	
Modifications of physical artifacts	Participants used several colors or tags to mark item. After moving an item to a separate list on a separate sheet of paper, the partipants crossed out the item on the original list.	

Fig. 3. Subsequent artifacts produced by GRE for shopping list example.

Würfel, Lutz & Diehl – Grounded requirements engineering: An approach to use case driven requirements engineering (The Journal of Systems and Software, 2016)

Übung 6


Days before OpenAI
Developer coding - 2 hours
Developer debugging - 6 hours
Days after OpenAI
ChatGPT generates Codes - 5 min
Developer debugging - 24 hours

Aufgabenstellung und Vorgehensweise

→ Übungsblatt in StudIP

Fragebogen: Programmieren mit ChatGPT

Frage 1: Wie verwenden Sie ChatGPT (oder andere LLMs) als Hilfe zum Programmieren?

Frage 2: Begründen Sie, warum Sie ChatGPT auf diese Weise verwenden?

Frage 3: Denken Sie an das letzte Mal, als Sie ChatGPT (oder anderes LLM) als Hilfe zum Programmieren eingesetzt haben. Wie sind Sie vorgegangen?

CAQDAS

- Computer-Assisted Qualitative Data Analysis Software

FEEDING YOUR QUALITATIVE NEEDS

https://gandalf.uni-trier.de:8443/

- Taguette (www.taguette.org)
  - **free and open source** tool for qualitative research
  - Collaborative
  - Highlight, tag and export results
  - Supports the researcher in importing data; creating, applying, and refining codes and categories

- MF – Michael Feldmann
- PG – Philipp Geier
- MH – Maxim Hotz
- JJ – Joel Jax
- HK – Hilal Khalife
- JK – Josua Kirsch
- JL – Jan-Niclas Loosen
- CP – Christoper Probst
- JR – Jan Niclas Ruppenthal
- JS – Jessica Schiffer
- FS – Fabian Sponholz
- SS – Simon Szulik
- RT – Raphael Thelen
- JW – Justin Weich
- MW1 – Malte Witt
- MW2 – Matthias Wölwer