Types of Cooperation Episodes in Side-by-Side Programming

Lutz Prechelt, Ulrich Stärk, Stephan Salinger
Institut für Informatik
Freie Universität Berlin

- What is side-by-side prog.?
  - And why are we interested?
- Research motivation
- Research question
- Study setup
  - Recordings of SbS sessions
  - Grounded Theory
- Results

What is Side-by-Side Programming (SbS)?

Freie Universität Berlin

Pair programming:

2 people
1 computer
1 task

Side-by-side programming:

2 people
2 computers
1-2 tasks

http://blog.touristr.com/2009/03/08/aov-day-4

Close physical proximity
Switch work modes as appropriate

What is Side-by-Side Programming (SbS)?

Freie Universität Berlin

In other words:

Two people working together,

two computers, oh, and they never

move far apart,

"To help is an art.",

so they do now-and-then, not forever.

Why are we interested in SbS?

- Pair Programming (PP) has many potential advantages:
  - productivity, quality, learning, focus, broader ownership, satisfaction, etc.
- and some potential disadvantages:
  - may be too intense to use it always (at least for some people)
  - is boring and wasting for simple tasks

- Side-by-side programming attempts to get most benefits of PP while avoiding its drawbacks
  - suggested by Alistair Cockburn: Crystal Clear, Addison-Wesley, 2004.

Why are we interested in SbS?

Freie Universität Berlin

In other words:

Pair Programming is fine

for quality, learning, and time,

but it can be boring

or overly soring,

so try Side-by-side 5 til 9.

Related work 1

- Jerzy R. Nawrocki, Michal Jasinski, Lukasz Olek, Barbara Lange: Pair Programming vs. Side-by-Side Programming.
  Lecture Notes in Computer Science, 3792:28-38, 2005.

- Controlled experiment. Compares time-to-finish-task for solo programmers, PP pairs, and SbS pairs

- Time to finish programming task:
  Solo 100%, PP 74%, SbS 61%

- Less knowledge of overall source code for SbS than PP or solo
  - Conjecture from change task observations

Related work 1

In other words:

Researchers expect SbS

more efficient than PP (an excess!)

And what do they gather

when they start to measure?

It is, by far! And no less!

Research motivation

- So far, most results on PP (dozens) and SbS (2) provide rather little insight
  - mostly quantitative, black-box
  - mixed results, but cannot explain differences

- Our overall research perspective:
  - Understand the actual processes of PP and SbS
    - by qualitative analysis (Grounded Theory Method)
  - in order to formulate constructive advice on their use
    - by process patterns and process anti-patterns
  - and obtain means for measuring the hard-to-quantify aspects

Research motivation

In other words:

We want to describe what they do

when people pair up as a crew.

Want to see what goes well

and what goes to hell,

to advise, to make promises true.

Research question

- When and why and for what purpose
  do side-by-side programmers cooperate directly?
  - I.e., when/why do they use pair mode
    as opposed to solo mode?

(Our results focus on the purpose)

Research question

In other words:

When and why, for what end

do Side-by-Side partners bend

their attention aside

for helping their bride

and hear "Thank you, that was heaven-sent!"

Study setup: Recordings of SbS sessions

Freie Universität Berlin

- 4-day workshop on Java web development
  - packed full with technical content: Hibernate, Spring, Tapestry
  - 10 participants (senior students), working in teams of two
  - daily practice sessions, leading to a small application
- Setup and tasks were such that teams practiced SbS
  - without ever being taught or told to
- We recorded the last session of each of three teams as follows
  - 2x desktop video
  - 2x webcam-on-top-of-monitor video
  - 2x audio
  - each session had about 2.5 hours length
    - 7.5 hours overall → very much for a Grounded Theory analysis
- Participants then answered a postmortem questionnaire

Study setup:
Recordings of SbS sessions

Freie Universität Berlin

In other words:

After 4 days of web-program workshop

our three teams did one more task in pair-hop

A 7-hour ode

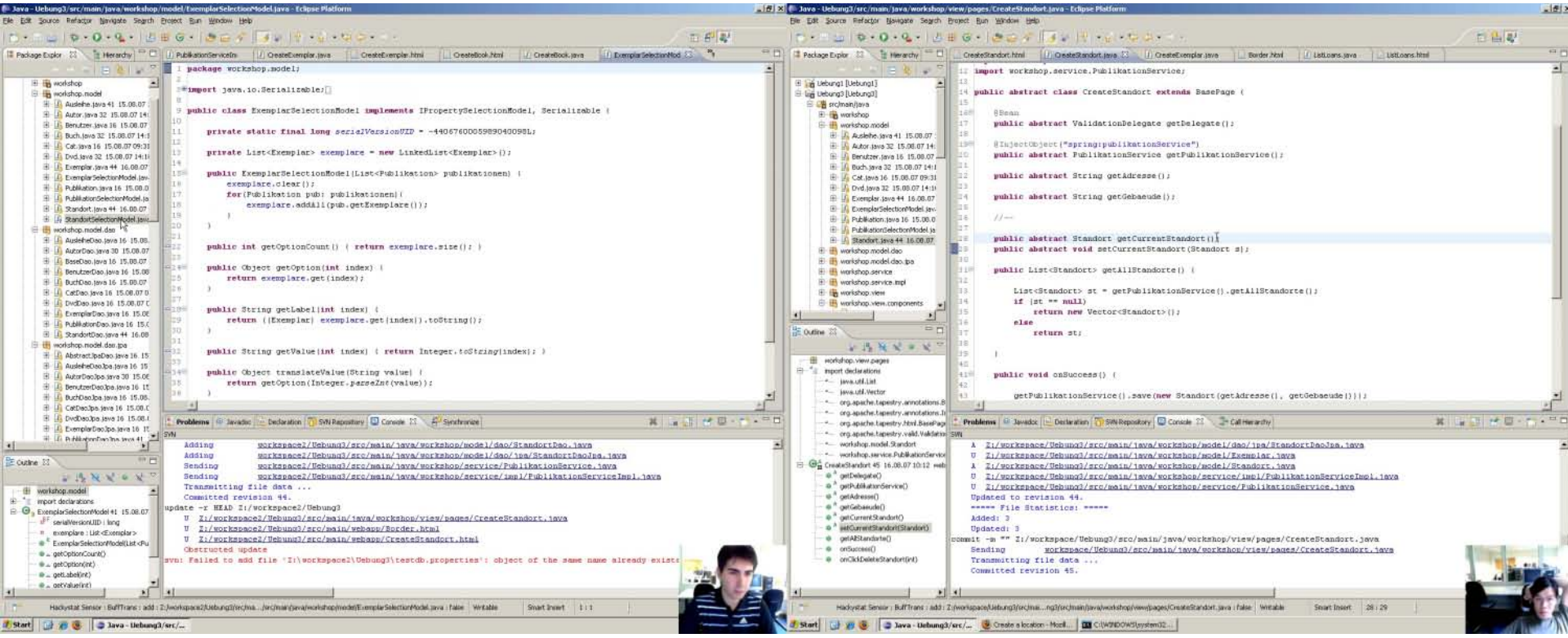
in 6-channel mode

we recorded. That's too much, so do crop, CROP!

Combined data source for analysis

Freie Universität Berlin

For each session,
we combined all 6 channels into a single 2560-pixel-wide video



Combined data source for analysis

In other words:

There are some slides that are really hard

to put in a Limerick Format

This is one of these

so please stay at ease

if it is not quite as informative and as well-structured as the others

Data analysis: Grounded Theory

Freie Universität Berlin

- In a nutshell, Grounded Theory (GT) means
  - conceptualizing observations abductively (i.e. by bringing in spontaneous ideas)
  - validating and refining the concepts via constant comparison
  - and observing relationships in order to arrive at a theory
- GT is known to be extremely time-consuming
- With data as rich as ours, this is even worse
  - "drowning in observations"

- ➔ Use the *Foundation Layer* concepts as a starting point
  - developed from and for PP sessions
  - primarily concepts describing verbal interaction events
    - (presented at PPIG 2008)
  - Used as a set of candidate ideas (rather than prescriptively)
    - ➔ compatible with the GT approach
  - (example will follow)

Data analysis: Grounded Theory

In other words:

To form Grounded Theory takes long

before your concepts grow strong.

A Foundation Layer

of concepts 's a player

whose usage you will find not wrong.

Analysis procedure (simplified)

Freie Universität Berlin

1. **Identify** cooperation episodes
   - via surface phenomena:
     pairing with each other, talking to each other
2. **Conceptualize** episodes via Foundation Layer concepts
   - (example will follow)
3. Introduce additional concepts
   - describing properties of specific Foundation Layer concepts
   - support making important distinctions
   - (examples will follow)
4. **Cluster** similar episodes and conceptualize their similarities
   - employing a visualization as a support tool

111 episodes (of lengths 5 seconds to 31 minutes)

Analysis procedure (simplified)

Freie Universität Berlin

In other words:

The episodes are easy to free.

Conceptualize them with glee.

In each difficult spot

add a concept ad-hoc;

then cluster – and check what you see.

Example: Foundation Layer concepts and episode visualization

Example: Foundation Layer concepts and episode visualization

Freie Universität Berlin

In other words:

Proposing the next steps to make

then discussing what else is at stake

back and forth til "agree"

that's "Discuss Strategy",

one of seven such types that you'll see.

Oops.

The additional concepts

Mostly properties:

- *explicitness*(strategy):
  - procedural, declarative
- *granularity*(design), *granularity*(rationale):
  - coarse-grained, fine-grained
- *specificity*(knowledge):
  - project-specific, generic

- *type-of*(propose_step):
  - help me, help you, stop help me, stop help you
- *type-of*(knowledge):
  - description of phenomenon, explanation for phenomenon, other
- *outcome*(verify_something):
  - correct, incorrect, don't know

The additional concepts

In other words:

Explicitness of strategy,

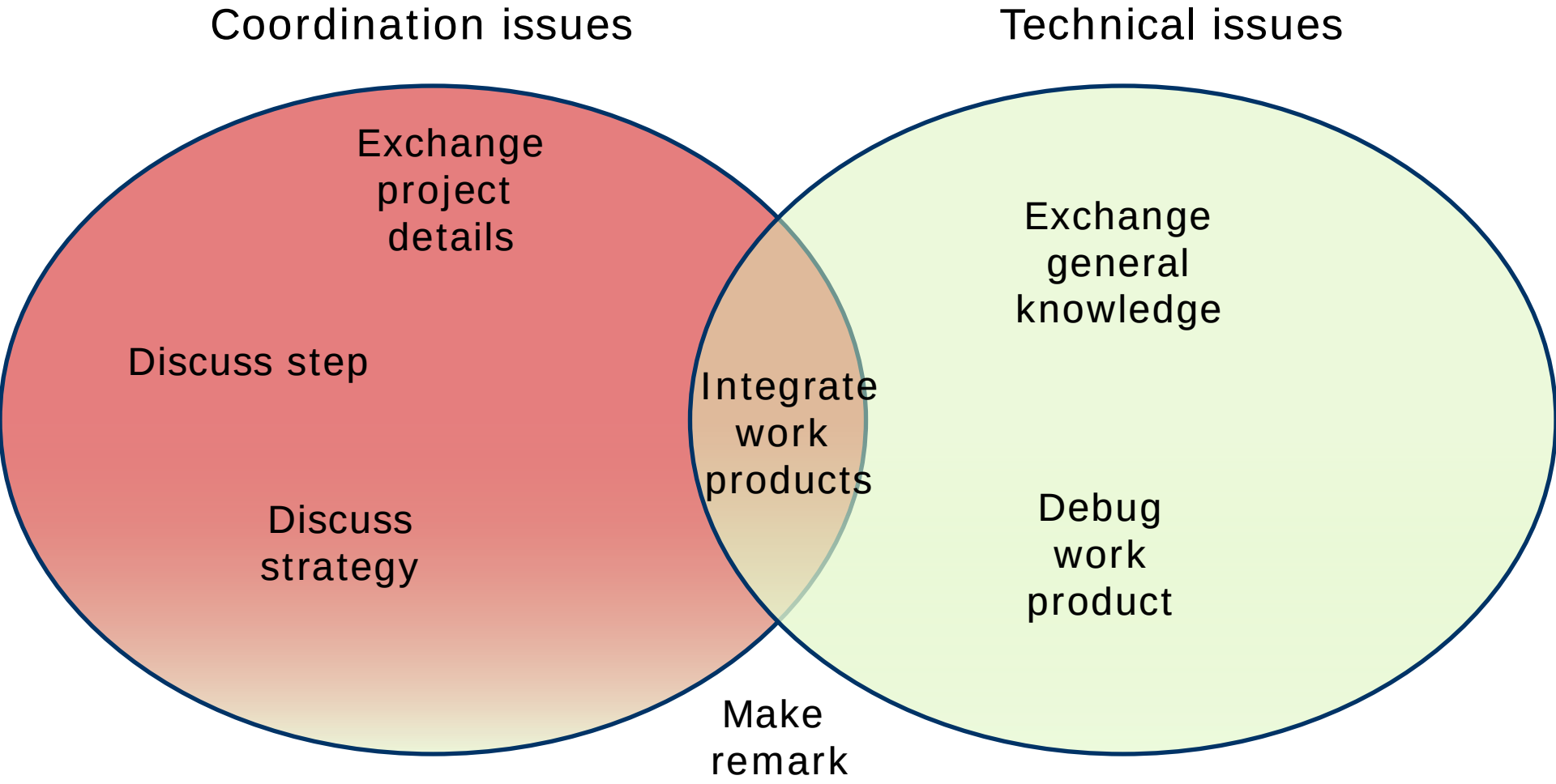
procedural de-clarity,

generic, specific,

that's really terrific

you're totally confusing me.

Results: Cooperation episode types



Results: Cooperation episode types

In other words:

Here you saw two coloured rounds

with labels (imperative nouns).

Now, that wasn't hard

so don't be a fart

and affirm how IYFA that sounds.

"Insert Your Favourite Adjective": simple|impressive|scientific

Cooperation episode types: Coordination issues

Freie Universität Berlin

- **Exchange project details**
  - Partners inform each other about
    - current work status (concepts: *_*completion*, *_*state*),
    - design facts etc. (concepts: *_*knowledge*$_{project\_specific}$), or
    - background inform. (concepts: *_*rationale*)

    relevant only within this project/task

- **Discuss step**
  - purpose: fine-grained work planning
  - concepts: *_step, *_strategy$_{procedural}$, design$_{fine_grained}$, *_rationale$_{fine_grained}$, *_todo

- **Discuss strategy**
  - purpose: coarse-grained work planning
  - concepts: *_strategy$_{declarative}$, design$_{coarse_grained}$, *_rationale$_{coarse_grained}$,

Cooperation episode types: Coordination issues

Freie Universität Berlin

In other words:

Coordination means to agree

what to do (short- or long-term), you see

Or describe your work status

as a "Partner, update us!"

and that's all that we found there to be.

Cooperation episode types: Technical issues

Freie Universität Berlin

- **Exchange general knowledge**
  - One answers a query of the other, typically regarding technology, libraries, tools, etc.
  - concepts: $\text{ask\_knowledge}_{\text{generic}}$, $\text{explain\_knowledge}_{\text{generic}}$
- **Debug work product**
  - One helps the other finding a defect
  - …either after a query ($\text{propose\_step}_{\text{help\_me}}$, $\text{ask\_knowledge}_{\text{explanation\_for\_phenomenon}}$)
  - …or spontaneously ($\text{propose\_step}_{\text{help\_you}}$, $\text{ask\_knowledge}_{\text{description of phenomenon}}$)

Cooperation episode types: Technical issues

Freie Universität Berlin

In other words:

In the technical realm you explain

tools and libraries, new or arcane,

or are helping detect

13 bugs, 1 defect

the rest is like solo, the same.

Cooperation episode types: Other

Freie Universität Berlin

Both coordination and technical at once:

- **Integrate work products**
  - Partners put together pieces each has developed alone
  - Complex episodes, consisting of status check, decision, sync, test, and possibly debug
    - status check is an "Exchange product details" episode
    - decision is a "Discuss step" episode
    - debug is a "Debug work product" episode

Other:

- **Make remark**
  - One partner spontaneously comments the work of the other
    - possible only because of the "osmotic communication" in SbS
  - We only saw two instances: propose_step, explain_knowledge

Cooperation episode types: Other

Freie Universität Berlin

In other words:

When putting the pieces together,

they flock like birds of a feather.

When your partner is stark

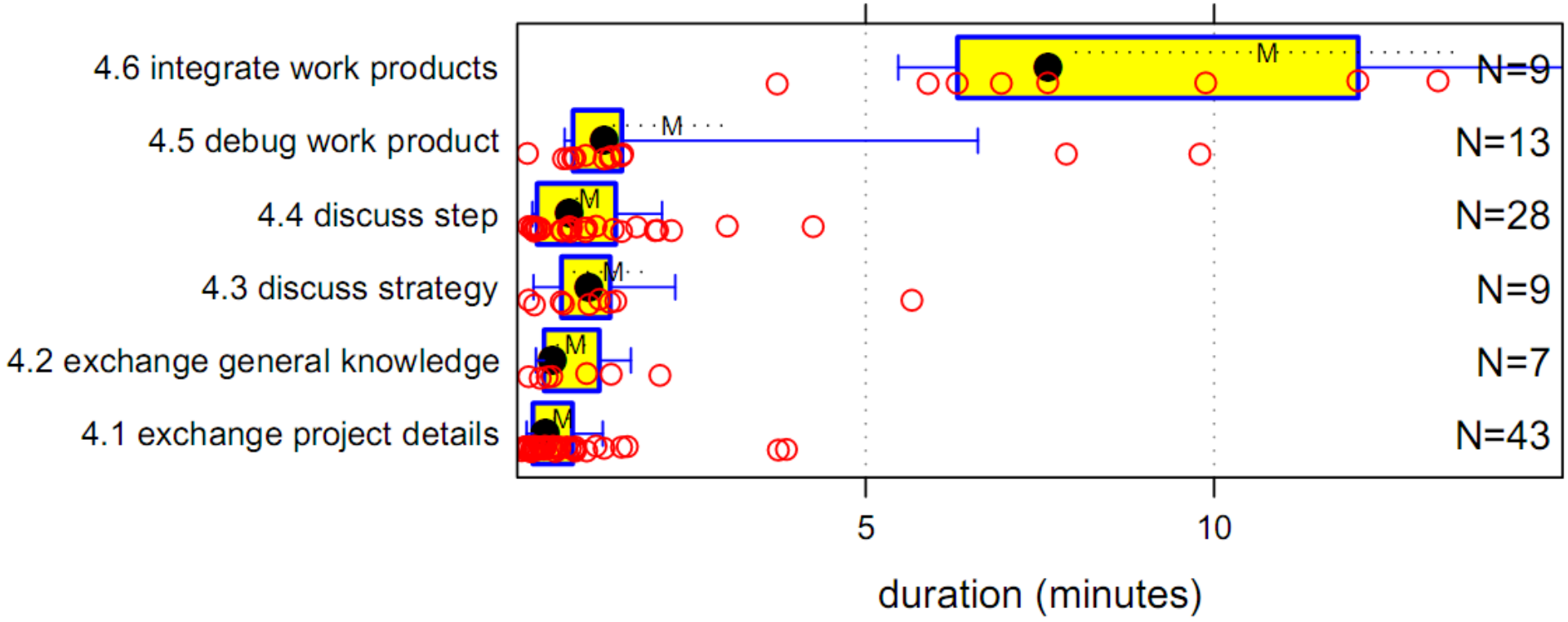
you may Make a Remark

to improve his work results' tether.

Duration and frequency of episode types

Freie Universität Berlin

- Coordination issues are by far more frequent



Duration and frequency of episode types

Freie Universität Berlin

In other words:

You coordinate most of the times

but that doesn't account for the dimes:

Integrate quicker!

That's the real picker!

That's were solo work's paying its fines.

Threats to validity

Freie Universität Berlin

Internal validity

- Use of Foundation Layer:
  - a mere cost saver
  - distortion is unlikely
- Wrong types derived?:
  - partitioning into types is somewhat arbitrary
    - coarser or finer grain possible
  - Our types are certainly real (due to use of GT):
    - "If they make sense to you, they are also valid"

External validity

- Student subjects, lab task:
  - Frequencies may differ with professionals
  - Further types may exist
  - But our types ought to be ecologically valid
- Only three sessions:
  - Further types may exist

Threats to validity

In other words:

That topic is really too serious

to summarize it by delirious

rhymes of some sort.

So I will abort

my Limerick here sound-research-erious.

Thank you!

Thank you!

Freie Universität Berlin

In other words:

I hope that the rhymses' meandering

has not too much reduced understandering

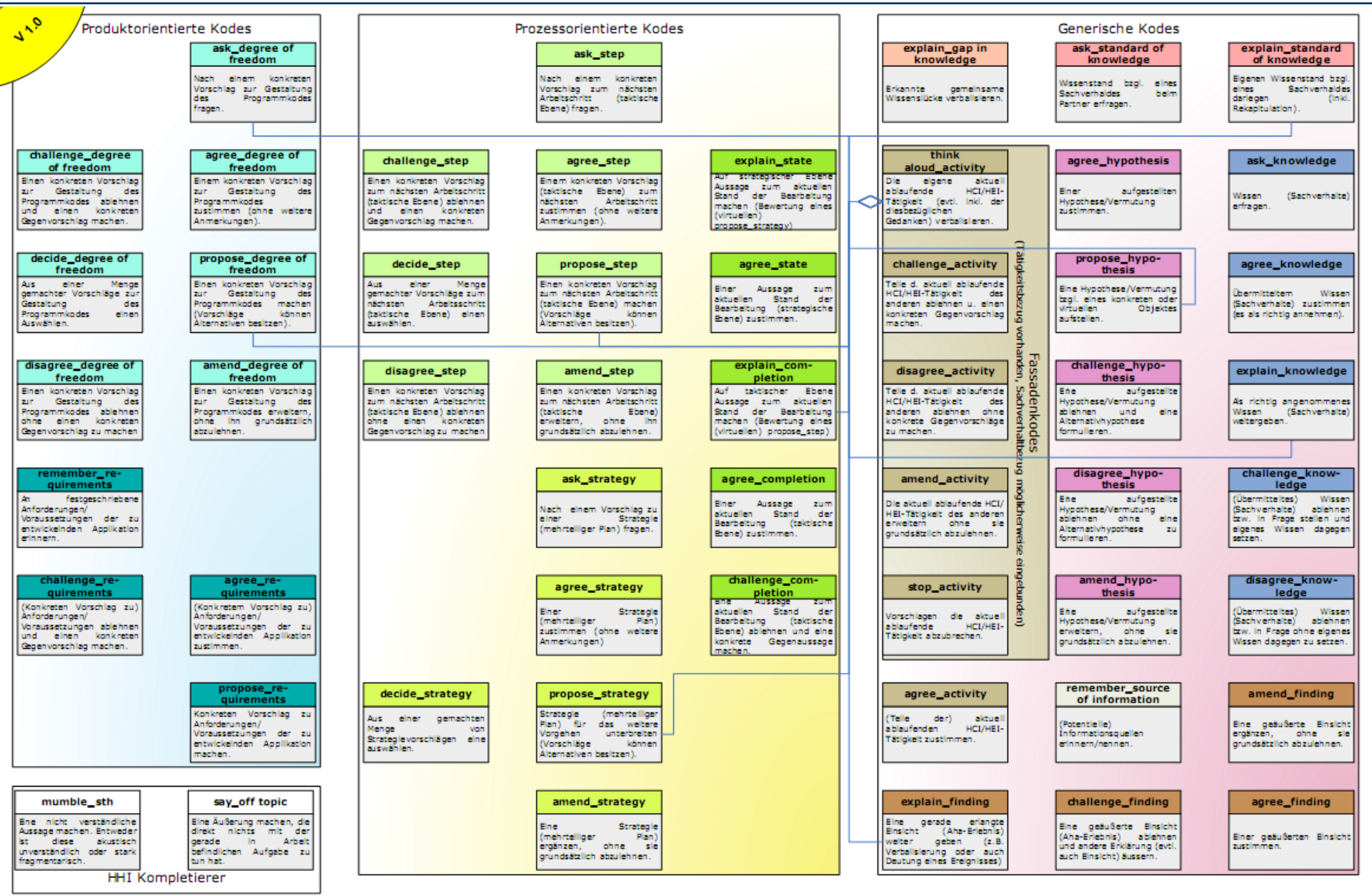
If it has: I am sorry,

I'll jump onto a lorry

leaving Limerick right now, regretendering.

Are there still any questions?

The Foundation Layer (excerpt)



The subjects

	pair 1		pair 2		pair 3	
Gender (male/female)	m	m	m	m	f	m
Been a student since (no. of terms)	14	12	6	8	7	8
Java programming experience (years)	6	7	3	4	1	4
Java web development experience (years)	1	0	2	1	1	2
I am among the most capable x%	40%	5%	40%	40%	50%	40%
Quality of cooperation (1–5)[1]	4	3	5	4	5	4
Task difficulty (1–5)[2]	3	2	3	3	4	3

[1] 1: very bad, 3: OK, 5: very good

[2] 1: much too easy, 3: just right, 5: much too difficult

Related work 2

Dewan et al.: Distributed Side-by-Side programming

Identifies a number of surface-level "work modes":

- Concurrent uncoupled programming
  - solo mode
- Concurrent coupled programming
  - work separately, but talk to each other
- Pair programming
- Concurrent programming/browsing
  - like PP, but observer investigates additional material
- Concurrent browsing

Largely orthogonal to our cooperation episode types.