

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)?



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)?



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.





- 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.





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 <u>actual processes</u> 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



- 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



In other words:

After 4 days of web-program workshop

our three teams did one more task in pair-hop

A 7-hour ode

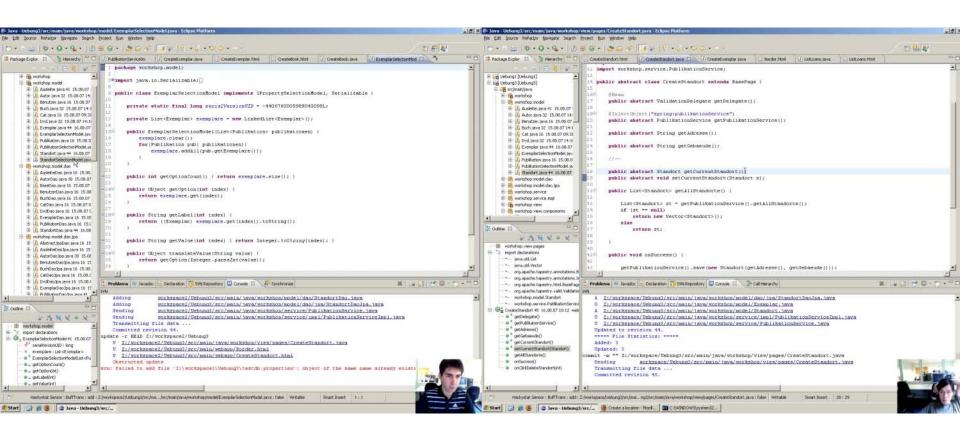
in 6-channel mode

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



Combined data source for analysis

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







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



- 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"
- Juse 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)
 - accompatible with the GT approach
 - (example will follow)





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)



- 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)





In other words:

The episodes are easy to free.

Conceptualize them with glee.

In each difficult spot

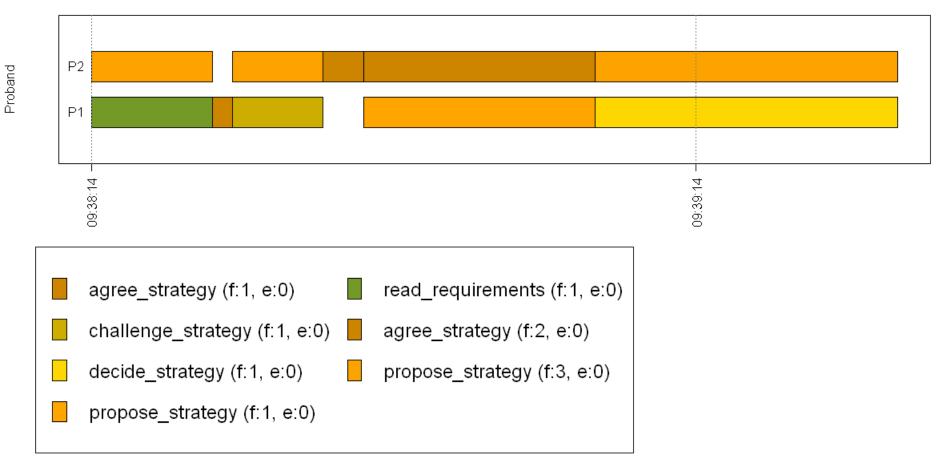
add a concept ad-hoc;

then cluster – and check what you see.

Example: Foundation Layer concepts and episode visualization



Pair 2, "Discuss strategy" episode



Example: Foundation Layer concepts and episode visualization



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

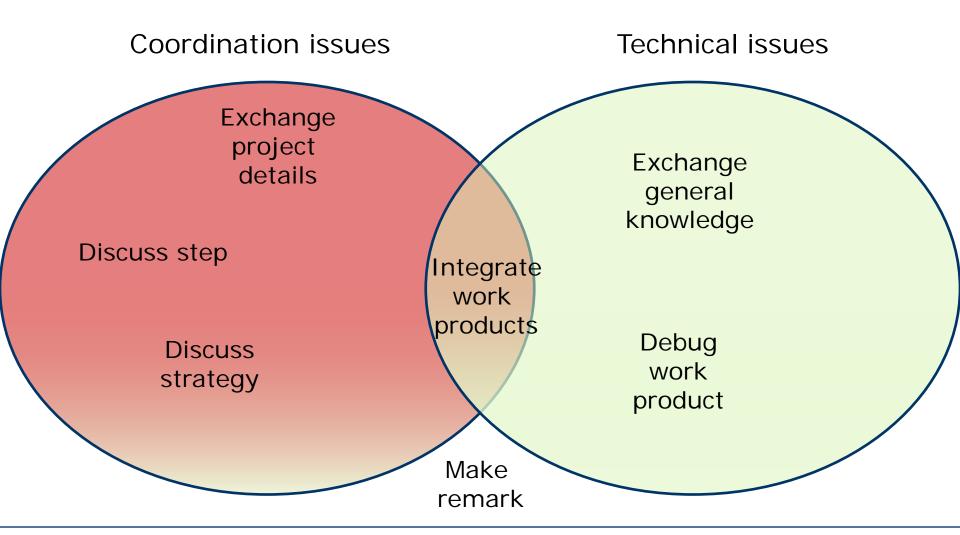


In other words:

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



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



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



Exchange general knowledge

- One answers a query of the other, typically regarding technology, libraries, tools, etc.
- concepts: ask_knowledge_{generic}, explain_knowledge_{generic}

Debug work product

- One helps the other finding a defect
- ...either after a query (propose_step_{help_me}, ask_knowledge_{explanation_for_phenomenon})
- ...or spontaneously (propose_step_{help_you}, ask_knowledge_{description of phenomenon})

Cooperation episode types: Technical issues



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



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



In other words:

When putting the pieces together,

they flock like birds of a feather.

When your partner is stark

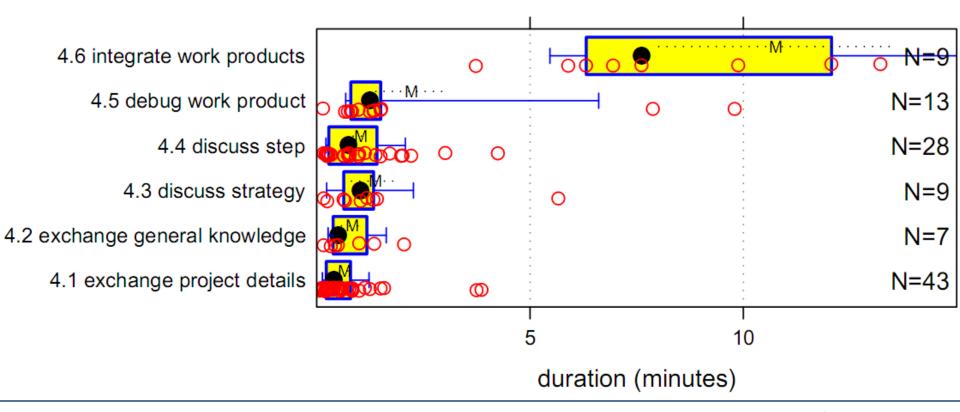
you may Make a Remark

to improve his work results' tether.

Duration and frequency of episode types



Coordination issues are by far more frequent



Duration and frequency of episode types



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



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!



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)

HHI Kompletierer



779 Produktorienti			Prozessorientierte Kodes				Generische Kodes	
	ask_degree of freedom		ask_step			explain_gap in knowledge	ask_standard of knowledge	explain_standard of knowledge
	Nach einem konkreten Vorschlag zur Gestaltung des Programmkodes fragen.		Nach einem konkreten Vorschlag zum nächsten Arbeitschritt (taktische Ebene) fragen.			Erkannte gemeinsame Wissenslücke verbalisieren.	Wissenstand bzgi. eines Sachverhaldes beim Partner erfragen.	Elgenen Wissenstand bzgl. eines Sachverhaldes darlegen (Inkl. Rekapitulation).
challenge_degree	agree_degree of					think		
of freedom Binen konkreten Vorschlag zur Gestaltung des Programmkodes ablehnen und einen konkreten Gegenvorschlag machen.	freedom Einem konkreten Vorschlag zur Gestelbung des Programmkodes zustimmen (ohne weltere Anmerkungen).	Challenge_step Binen konkreten Vorschlag zum nächsten Arbeitschritt (taktische Bbene) ableinnen und einen konkreten Gegenvorschlag machen.	agree_step Einem konkreten Vorschlag (taktische Ebene) zum nächsten Arbeitschritt zustimmen (ohne weitere Anmerkungen).	explain_state Aur strategister Ebene Aussage zum aktuellen Stand der Bearbeitung machen (Bewertung eines (virtuellen) propose_strategy)	~	aloud_activity Die eigene aktuell ablaufende HCI/HEI- YEtigkeit (evdl. inkl. der diesbezüglichen Gedanken) verballsieren.	agree_hypothesis Einer surgestellten Hypothese/Vermutung zustimmen.	wissen (Sachverhalte) erfragen.
decide_degree of freedom	propose_degree of freedom	decide_step Aus einer Menge	propose_step Einen konkreten Vorschlag	agree_state		challenge_activity Telle d. sktuell ablaufende	propose_hypo- thesis	agree_knowledge
gemachter Vorschläge zur Gestaltung des Programmkodes einen Auswählen.	zur Gestaltung des Programmikodes machen (Vorschläge können Alternativen besitzen).	gemachter Vorschläge zum nächsten Arbeitsschritt (taktische Ebene) einen auswählen.	zum nächsten Arbeitschritt (taktische Ebene) machen (Vorschläge können Alternativen besitzen).	aktuellen Stand der Bearbeitung (strategische Bbene) zustimmen.		Telle d. siktuell ablaufende HCI/HEI-Tätigkelt des anderen ablehnen u. einen konkreten Gegenvorschlag machen.	trgl. eines konkreten oder virtuellen Objektes aufstellen.	Übermitteltem Wissen (Sachverhalte) zustimmen (es als richtig annehmen).
disagree_degree of	amend_degree of	disagree_step	amend_step	explain_com-		hander Fas	challenge_hypo-	explain_knowledge
freedom Binen konkreten Vorschlag zur Gestaltung des Programmkodes ablehnen ohne einen konkreten Gegenvorschlag zu machen	freedom Binen konkreten Vorschlag zur Gestelbung des Programmkodes erweitern, ohne ihn grundsätzlich abzulehnen.	Einen konkreten Vorschlag zum nächsten Arbeitschritt (taktische Ebene) ablehnen ohne einen konkreten Gegenvorschlag zu machen	Binen konkreten Vorschlag zum nächsten Arbeitschritt (taktische Bbene) erweitern, ohne ihn grundsätzlich abzulehnen.	Auf taktischer Ebene Aussage zum aktuellen Stand der Bearbeitung machen (Bewertung eines (virtuellen) propose_step)		disagree_activity Telle d. sktuell ablaufende HCI/HEI-Tatigket des anderen ablehnen ohne konkrete Gegenvorschläge zu machen.	thesis Ene sufgestellte Hypothese/Vermutung sblehnen und eine Alternativhypothese formulieren.	As richtig angenommenes Wissen (Sachverhalte) weitengeben.
romember re-						tbezug	disagree_hypo-	challenge_know-
remember_re- quirements An festgeschriebene Anforderungen/ Voraussetrungen der zu entwickelnden Applikation erinnern.			ask_strategy Nach einem Vorschlag zu einer Strategle (mehrteiliger Plan) fragen.	agree_completion Einer Aussage zum sktuellen Stand der Bearbeitung (taklische Bene) zustimmen.		amend_activity Die aktuell ablaufende HCI/ HEI-Tätigkeit des anderen enweitern ohne sie grundsätzlich abzulehnen.	thesis Ehe aufgestellte Hypothese/Vermutung ablehnen ohne eine Alternativhypothese zu formulieren.	(Übermitteltes) Wissen (Sachverhalte) ablehnen tzw. In Frage stellen und eigenes Wissen dagegen setzen.
challenge_re- quirements (Konkreten Vorschlag zu) Anfordenunger/ Vorsussetzungen ablehnen und einen konkreten Gegenvorschlag machen.	agree_re- quirements (Konkretem Vorschlag zu) Anforderungen/ Vorsussetzungen der zu entwickeinden Applikation zustimmen.		agree_strategy Einer Strategie (mehrteiliger Plan) zustimmen (ohne weltere Anmerkungen)	challenge_com- pletion me Aussage zum situtellen Stand der Bearbeitung (taktische Ebene) ablehnen und eine konkrete Gegenaussage machen.		stop_activity Vorschlagen die aktuell ablaufende HCI/HEI-Tätigkeit abzubrechen.	amend_hypo- thesis Ehe surgestellte Hypothese/Vermutung enweltern, ohne sie grundsétzilich abzulehnen.	disagree_know- ledge (Übermitteltes) Wissen (Sachverhalte) ablehnen tzw. In Frage ohne eigenes Wissen dagegen zu setzen.
	propose_re- quirements	decide_strategy	propose_strategy			agree_activity	remember_source	amend_finding
	Konkreten Vorschlag zu Anforderungen/ Voraussetzungen der zu entwickelnden Applikation machen.	Aus einer gemachten Menge von Strategie vorschlägen eine auswählen.	Strategie (mehrteiliger Plan) für das weitere Vorgehen unterbreiten (Vorschläge können Alternativen besitzen).			(Telle der) aktuell ablaufenden HCI/HEI- Tätigkelt zustimmen.	(Potentielle) Informationsquellen erinnern/nennen.	Eine geäußerte Einsicht ergänzen, ohne sie grundsätzlich abzulehnen.
mumble_sth Bne_nicht_verständliche	say_off topic Eine Äußerung machen, die		amend_strategy Eine Strategie			explain_finding Eine gerade erlangte	dhallenge_finding	agree_finding
Aussage machen. Entweder ist diese akustisch unverständlich oder stark fragmentarisch.	direkt nichts mit der gerade in Arbeit befindlichen Aufgabe zu um hat.		(mehrteiliger Plan) ergänzen, ohne sie grundsätzlich abzulehnen.			Binsicht (Aha-Erlebnis) welter geben (z.B. Verballsierung oder auch Deutung eines Ereignisses)	(Aha-Erlebnis) ablehnen und andere Erklärung (evtl. auch Einsicht) äussern.	Einer geäußerten Einsicht zustimmen.

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

¹1: very bad, 3: OK, 5: very good

²1: 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.