Observations on Knowledge Transfer of Professional Software Developers during Pair Programming

Franz Zieris  
zieris@inf.fu-berlin.de  

Lutz Prechelt  
prechelt@inf.fu-berlin.de
Typical pair programming scenarios

1. Solve a difficult problem
   - Combined ideas
   - Combined background knowledge

2. Introduce new team member
   - System understanding
   - Common practices, coding conventions
Research Goal

Characterize **how** effective and efficient knowledge transfer works in pair programming.

• Geared towards practitioners:
  – Formulate **patterns** of beneficial and problematic behavior
Data collection

Authentic
- In-vivo recordings of professionals: no artificial pairs, tasks, or settings

Diverse
- Different domains & pair types
Data analysis

49+ Sessions from 11 companies
13 Sessions selected

Lord of the Rings I-III
Data analysis

13 Sessions selected

Grounded Theory (Strauss & Corbin)

450 Knowledge Transfer Episodes analyzed
Three Examples
Example 1: Efficient Pull Episode

Context: Java, with complex self-written framework
Task: Get business object (“MiniObject”) from proxy object (“objectHandle”)

```
return objectHandle.fetchM;
```

- `fetchMiniObject()` 
- `CMMiniObject`
- `fetchMiniObject(boolean allowMicroObject)`

Do the MicroObjects still exist?

Bob

Carl
Example 1: Efficient Pull Episode

Context: Java, with complex self-written framework
Task: Get business object (“MiniObject”) from proxy object (“objectHandle”)

```java
return objectHandle.fetchMiniObject(true);
```

Do the MicroObjects still exist?

Yes, they do. They do here, on your machine. On the working branch, however ... true.
Example 1: Efficient Pull Episode

Context: Java, with complex self-written framework
Task: Get business object (“MiniObject”) from proxy object (“objectHandle”)

```java
return objectHandle.fetchMiniObject(true);
```

Do the MicroObjects still exist?

Yes, they do. They do here, on your machine. On the working branch, however ... true.

Bob

Carl
Example 2: Effective Resynchronization
Example 2: Effective Resynchronization
Example 2: Effective Resynchronization
Example 2: Effective Resynchronization
Example 2: Effective Resynchronization
Example 2: Effective Resynchronization
Example 2: Effective Resynchronization

**Co-Produce Episode**

**Design Proposal**

**Push Episode**

**Internal Knowledge Need**

**Talking Pioneer Episode**

**External Knowledge Need**

RESYNC
Problematic Example 3: Parallel Work

Parallel Production Episode

Ah-ha!

Ah.
Problematic Example 3: Parallel Work

Parallel Production Episode

Target Content: 0

Ah-ha!

Ah.

TC: ?

Target Content: 0

TC: ?
Problematic Example 3: Parallel Work

Parallel Production Episode

TC: ?

Ah-ha!

Ah.

40 min

Code Change

Uh?

Internal Knowledge Need
Problematic Example 3: Parallel Work

Parallel Production Episode

Code Change

40 min

Internal Knowledge Need

Pull Episode

Target Content: 0

Ah-ha!

Ah.

Uh?

Target Content: 0

Target Content: 0

TC: ?

TC: √

TC: √

TC: √
Problematic Example 3: Parallel Work

Researcher’s perspective:
E’s confusion makes the effects of the Parallel Production Episode visible.
Summary of the examples

• Knowledge transfer episodes can be very efficient (with very few utterances)
  – ... if the pair **invests** in staying close together.

• If they don’t, episodes can become much **longer** and may take **several attempts**
  – See our previous work [1]

• Even worse, the pair risks working on **parallel** tracks (as in the last example)

Results

On Pair Programming in general
- No pair member more knowledgeable in all regards
- Positive effects of knowledge-wise inferior member

(Anti-)Pattern candidates
- Resync
- Parallel
- Talking Pioneering

Mechanisms of knowledge transfer episodes
- Push & Pull
- Co-Produce & Pioneering Production
Enough with Push ...
... time for Pull!
Thank you!

https://bitbucket.org/spooning
Used Images


https://www.flickr.com/photos/a2gemma/552208117


