# Investigating the Impact of Experience and Solo/Pair Programming on Coding Efficiency: Results and Experiences from Coding Contests

Dietmar Winkler[1] Martin Kitzler[2] Christoph Steindl[2] Stefan Biffl[1]

[1]Vienna University of Technology, Institute of Software Technology, Christian Doppler Laboratory "Software Engineering Integration for Flexible Automation Systems (CDL-Flex)", http://cdl.ifs.tuwien.ac.at

[2]Catalysts GmbH, Hagenberg, Austria
http://www.catalysts.cc

## Motivation & Goals

**„Yet another paper about pair programming!"** (reviewer comment)

Motivation:

- Pair programming is a well-investigated approach in agile development.
- Nevertheless, different findings on benefits of pair programming in literature requires additional empirical studies.
- Need for empirical studies and replications to strengthen the body of knowledge and to provide empirical evidence, e.g., by involving and attracting participants from industry.

Goals:

- Providing a flexible experiment environment that
  (a) attracts various groups of participants (including industry people).
  (b) enables easy replication of (large scale) experiments.
  (c) enables investigating effects of pair programming on performance.

Key research question focus on:

- How to provide a flexible experiment environment that attracts a large and heterogeneous group of participants to investigate the effects of pair programming?

## Solo & Pair Programming
### Related Work

Pair Programming (PP) aims at increasing software productivity at a higher level of software quality.

Based on various studies, a set of conclusions have been published, e.g.,

- PP supports learning in pairs in educational environments.
- PP in industry does not provide as extensive benefits as claimed by literature.
- Pairs that work independently are more productive that working concurrently.
- Expected benefits depends on the developer expertise and complexity of tasks.

→ Need for additional studies to investigate the effects of pair programming in different contexts.

→ How can we provide an experiment environment that supports empirical studies on pair programming?

# Coding Challenges and Contests
## Related Work

Empirical Studies and Controlled Experiments

- Controlled experiments are expensive and time-consuming.
- Challenging to attract (a) high numbers and (b) heterogeneous groups of participants (e.g., different experience levels, industry and academic people).
- Coding Challenges and Contest are promising candidates to overcome these limitations.

Some goals of coding contests

- Solve as much of given tasks as fast and as efficiently as possible[12].
- Identifying best coders in a group of organizations or countries[2].
- Recruiting contests organized by companies[36].
- Solving innovative tasks[4] or finding the best solution[5] for given problems.

[1] ACM International Collegiate Programming Contest (ICPC)
[2] International Olympiad in Informatics (IOI)
[3] Google CodeJam;
[4] Challenge24;
[5] TopCoder
[6] Catalysis Coding Contest

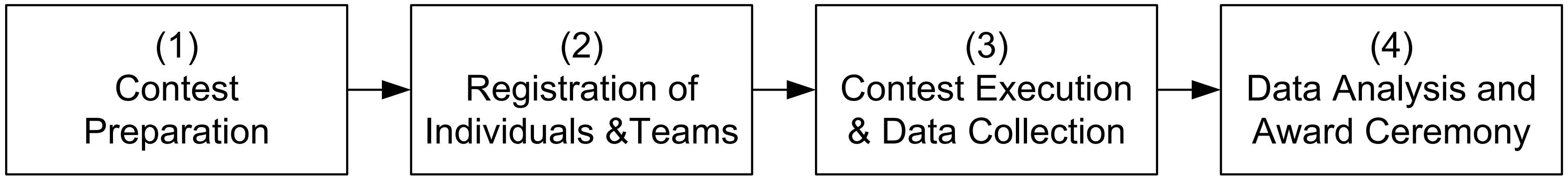
- We observed strong limitations regarding participants, tasks, development environments etc.

# Research Questions & Contest Process

Research Questions focus on

- Designing the catalysts coding contest as vehicle to supporting controlled experiments.
- Effects of solo/pair programming with respect to (a) application effort, (b) experiences levels, and (c) quality (i.e., number of defects).

Study Process



(1) Contest Preparation
(2) Registration of Individuals &Teams
(3) Contest Execution & Data Collection
(4) Data Analysis and Award Ceremony

(1) Contest Preparation: Contest environment setup and material preparation

(2) Preliminary registration of individuals and teams

(3) Contest Execution & Data Collection

(4) Analysis, Evaluation, and Award Ceremony

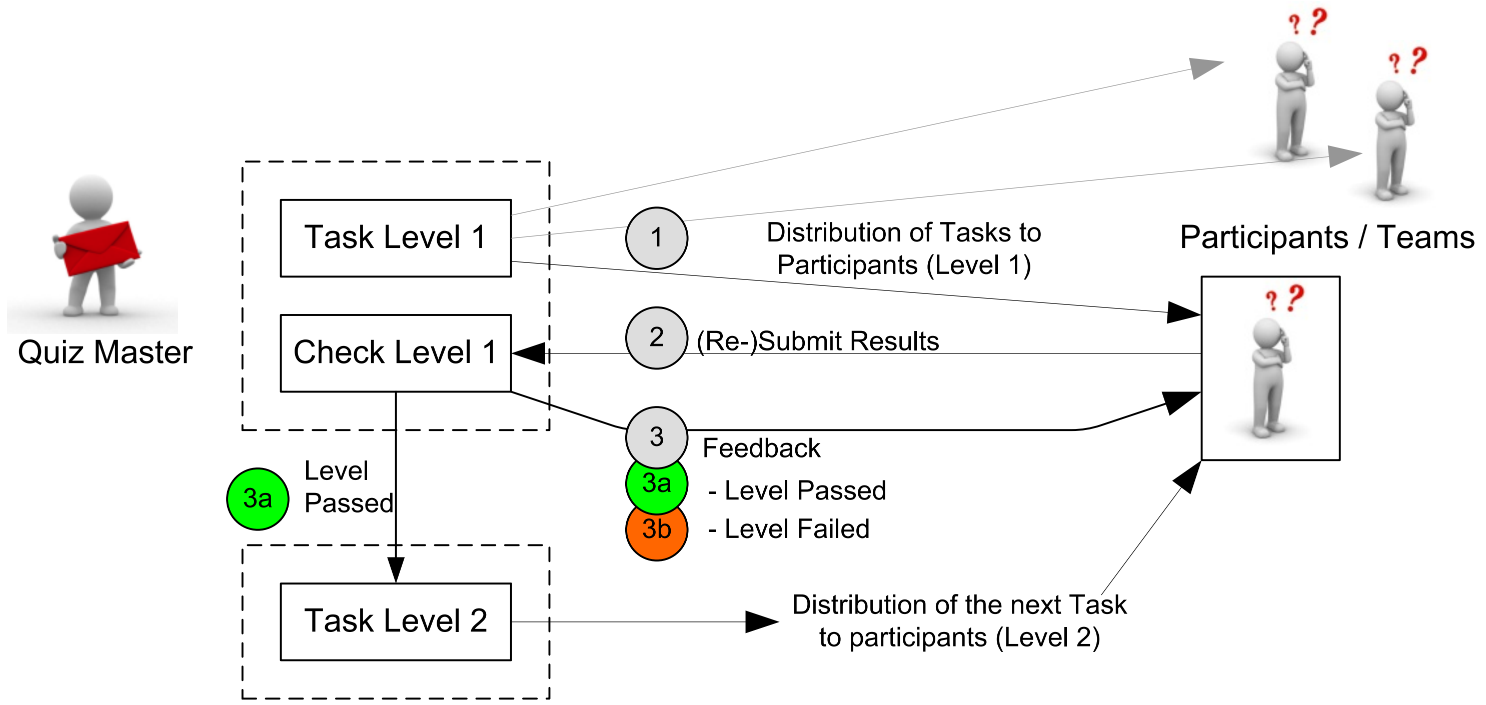
No restrictions regarding

- Programming language
- Experience
- Gender
- Nationality
- How tasks have to be solved

## Level-Based Contest/Study Environment

**Solving a set of defined tasks on 7 levels in sequence with increasing severity levels.**

1. Quiz Master hands out the first set of tasks to all participants.
2. Every participant group solves the tasks and submits the results as fast as possible.
3. Quiz Master checks the results and gives feedback
   (3a) results correct → positive feedback and next level
   (3b) incorrect results → negative feedback and next iteration on the same level.



## Contest/Study Setup

- Subjects: 95 participant groups, i.e., 53 solo programmers, 42 pairs → overall number of 137 individuals at three different experience levels:
  - juniors (up to undergraduate students)
  - seniors (experience programmers, typically graduate students), and
  - professionals from industry.
- Focus on prominent programming languages: Java (29.5%), C# (28.4%), C++ (42.1%)
- Time duration: upper time limit of 240 min
- Technical Infrastructure: Task submission, Results evaluation and feedback is provided by CatCoder (acting as "Quiz Master").
- Application. Lip reading program on 7 severity levels that calculates the most likely sentence that was formed by a number of input mouth shapes:
  - Lower levels (1-3) include the recognition of letters, words, syllables based on a dictionary holding 18k+ real English words.
  - Higher levels (4-7) includes pattern recognition based on a collection of Shakespeare texts including 551k+ words.
- Data Collection: Collection of communication data (CatCoder logfile).
- Data Analysis: Consistency checks of the data sets and statistical testing.

## Limitations

**Internal Validity:**

- Proven level-based contest environment based on a set of previous contests.
- Classroom setting to monitor and control study variables.
- Expert reviews and pilot tests of the application to verify correctness and feasibility
- Avoidance of communication between different participant groups.
- Experience questionnaire to capture skills of participants.

**External Validity**

- Limitation of the study duration 240 min
- Different experience levels have been captured

**Construction Validity**

- The study is based on related work and previous coding contests and addresses efficiency and duration, common variables in empirical studies

**Conclusion Validity**

- Statistical tests for hypothesis testing.

## Overview
### Results

- Overall number of **95 participant groups** at three different experience levels including: 53 solo programmers and 42 pairs.

| | Juniors | Seniors | Professionals | Total |
|---|---|---|---|---|
| # Participants | 24 | 51 | 20 | 95 |
| Overall Effort (avg) | 2:23 | 2:20 | 2:11 | 2:19 |
| Completed Levels (avg) | 1.8 | 2.6 | 2.2 | 2.3 |
| Completed Levels (max) | 4 | 7 | 3 | 7 |

- Comparable **overall effort** with focus on completed/passed levels.

- **Average number of Completed Levels:**
  - As expected juniors achieve on average the lowest level.
  - Maximum completed level: highest for seniors (7), followed by juniors (4) and professionals (3).

## Impact of Developer Experience
### Results

**Number of participants/pairs** per level

| # participants | Level 1 | Level 2 | Level 3 | Level 4 | Level 5 | Level 6 | Level 7 |
|---|---|---|---|---|---|---|---|
| Juniors | 24 | 13 | 4 | 2 | 0 | 0 | 0 |
| Seniors | 51 | 38 | 18 | 9 | 6 | 5 | 4 |
| Professionals | 20 | 17 | 6 | 0 | 0 | 0 | 0 |

**Average effort** per level (h:mm)

- Partly significant differences between Juniors and Seniors

| Avg Effort | Level 1 | Level 2 | Level 3 | Level 4 | Level 5 | Level 6 | Level 7 |
|---|---|---|---|---|---|---|---|
| Juniors | 1:17 | 1:32*[1] | 1:05 | 1:09*[2] | - | - | - |
| Seniors | 0:59 | 0:56*[1] | 1:11 | 0:43*[2] | 0:20 | 0:13 | 0:51 |
| Professionals | 0:57 | 1:09 | 0:50 | - | - | - | - |

**Average defects** per level

- Partly significant differences between Seniors and Professionals

| Avg Defects | Level 1 | Level 2 | Level 3 | Level 4 | Level 5 | Level 6 | Level 7 |
|---|---|---|---|---|---|---|---|
| Juniors | 2.9 | 1.31 | 1 | 1 | - | - | - |
| Seniors | 4.08*[3] | 1.89 | 2.56 | 1.22 | 1.83 | 0 | 1.75 |
| Professionals | 1.0*[3] | 1.18 | 1.83 | - | - | - | - |

## Programming Approach (Solo/Pairs)
### Results

**Number of participants/pairs** per level

- 1 pair and 3 individuals completed level 7.

| # participants | Level 1 | Level 2 | Level 3 | Level 4 | Level 5 | Level 6 | Level 7 |
|---|---|---|---|---|---|---|---|
| Individuals | 53 | 40 | 19 | 8 | 5 | 4 | 3 |
| Pairs | 42 | 28 | 9 | 3 | 1 | 1 | 1 |

Required **average effort** per level (h:mm)

- Benefits for individuals on level 1-3 and benefits for pairs on level 4-7.

| Avg effort | Level 1 | Level 2 | Level 3 | Level 4 | Level 5 | Level 6 | Level 7 |
|---|---|---|---|---|---|---|---|
| Individuals | 1:00 | 1:00 | 0:59 | 0:50 | 0:21 | 0:13 | 0:59 |
| Pairs | 1:06 | 1:14 | 1:20 | 0:44 | 0:13 | 0:14 | 0:28 |

**Average defects** per level

- No defects reported by pairs on level 4-7.

| Avg defects | Level 1 | Level 2 | Level 3 | Level 4 | Level 5 | Level 6 | Level 7 |
|---|---|---|---|---|---|---|---|
| Individuals | 3.2 | 1.7 | 1.8 | 1.6 | 2.2 | 0.0 | 2.3 |
| Pairs | 3.0 | 1.5 | 3.0 | 0.0 | 0.0 | 0.0 | 0.0 |

- No significant differences regarding effort and defects (up to 4 levels).

## Scoring and Ranking
### Results

**Award Ceremony - Final Scoring based on:**

- Maximum level reached.
- Time for level completion

**Findings**

- 3 pairs and 7 individuals are in the TOP-10, the **winner was a pair programming team**.
- 2 juniors and 8 seniors but **no professionals** are in the TOP-10

| Rank | Levels Completed | Total Defects Delivered | Programming Style | Experience Level |
|---|---|---|---|---|
| **1** | 7 | 0 | Pair | Senior |
| **2** | 7 | 7 | Solo | Senior |
| **3** | 7 | 9 | Solo | Senior |
| **4** | 7 | 10 | Solo | Senior |
| **5** | 6 | 12 | Solo | Senior |
| **6** | 5 | 23 | Solo | Senior |
| **7** | 4 | 13 | Solo | Senior |
| **8** | 4 | 6 | Solo | Junior |
| **9** | 4 | 13 | Pair | Senior |
| **10** | 4 | 4 | Pair | Junior |

## Summary & Future Work

**Summary:**

- Coding contests provide an organizational framework for planning/executing large-scale controlled experiments:
  - Involving heterogeneous groups of participants.
  - Foundation for (easy) study replication.
  - Capability to address various study objects and needs.
- Study on programming strategies (solo vs. pair programming) with interesting results:
  - No significant differences of average performance for different experience levels.
  - Seniors completed on average more levels but also reported more errors.
  - Pair programming tend to support more complex tasks (i.e., higher levels) and tend to deliver solutions at a higher level of quality.

**Future Work**

- Additional and in-depth analysis of the results.
- Analyzing the series of coding contests to improve the empirical evidence.
- Catalyst Coding Contest in October 2013 (http://www.catalysts.cc/alt/contest/?lang=en)

## Thank you ...

# Investigating the Impact of Experience and Solo/Pair Programming on Coding Efficiency: Results and Experiences from Coding Contests

Dietmar Winkler[1], Martin Kitzler[2], Christoph Steindl[2], Stefan Biffl[1]

[1] Vienna University of Technology, Institute of Software Technology, Christian Doppler Laboratory "Software Engineering Integration for Flexible Automation Systems" (CDL-Flex)

[2]Catalysts GmbH, Hagenberg, Austria

Dietmar.Winkler@tuwien.ac.at