Empirical evaluation of static type systems: A running experiment series

Stefan Hanenberg
University of Duisburg-Essen, Germany

Austin, Texas, 07.12.2012

This talk...

- Focus on „Whys“ not „Hows“ (London vs. Austin meeting)
  - No details about any experiment
  - Instead: Argumentation why certain steps in series have been done

Message of this task

- Do experiments

  => *...follow a rigorous method*

- Transitions between experiments are biased
  (as long as there is no clear theory)

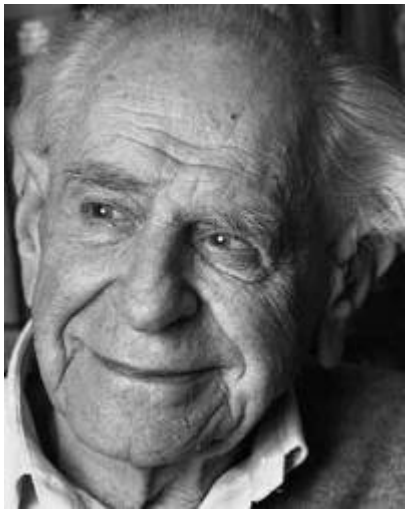
  => *...that's life, don't be afraid of it*

Starting Point (in 2006)

- Motivation
  - Giving measured arguments about type systems in teaching
  - „What is the impact of static type systems?“
- Starting point
  - Gannon'77, PrecheltTichy'98
  - Two small experiments, both show comparable results
- Ok, I want to do hypothesis testing

Empirical SE

- Following the approach of Karl Popper



  - Falsification of hypothesis (use of statically typed language decreases development time)

  - **NO PROOFS / NO GENERALIZABILITY**
    - But always the hope that repeated observations reveal some truth

Idealistic view on experimentation

- Well-known theory frequently tested

  Example: Galilei experiment
  Theory: *velocity of fall* depends on *mass*

- Falsification leads to reformulation of theory

- New theory
  - Stable against previous observations
  - New observation that falsified previous theory

What is a theory?

- Set of hypotheses, each consisting of implication relationships

  If stone A has m= 2 kg and stone B m = 4 kg, then

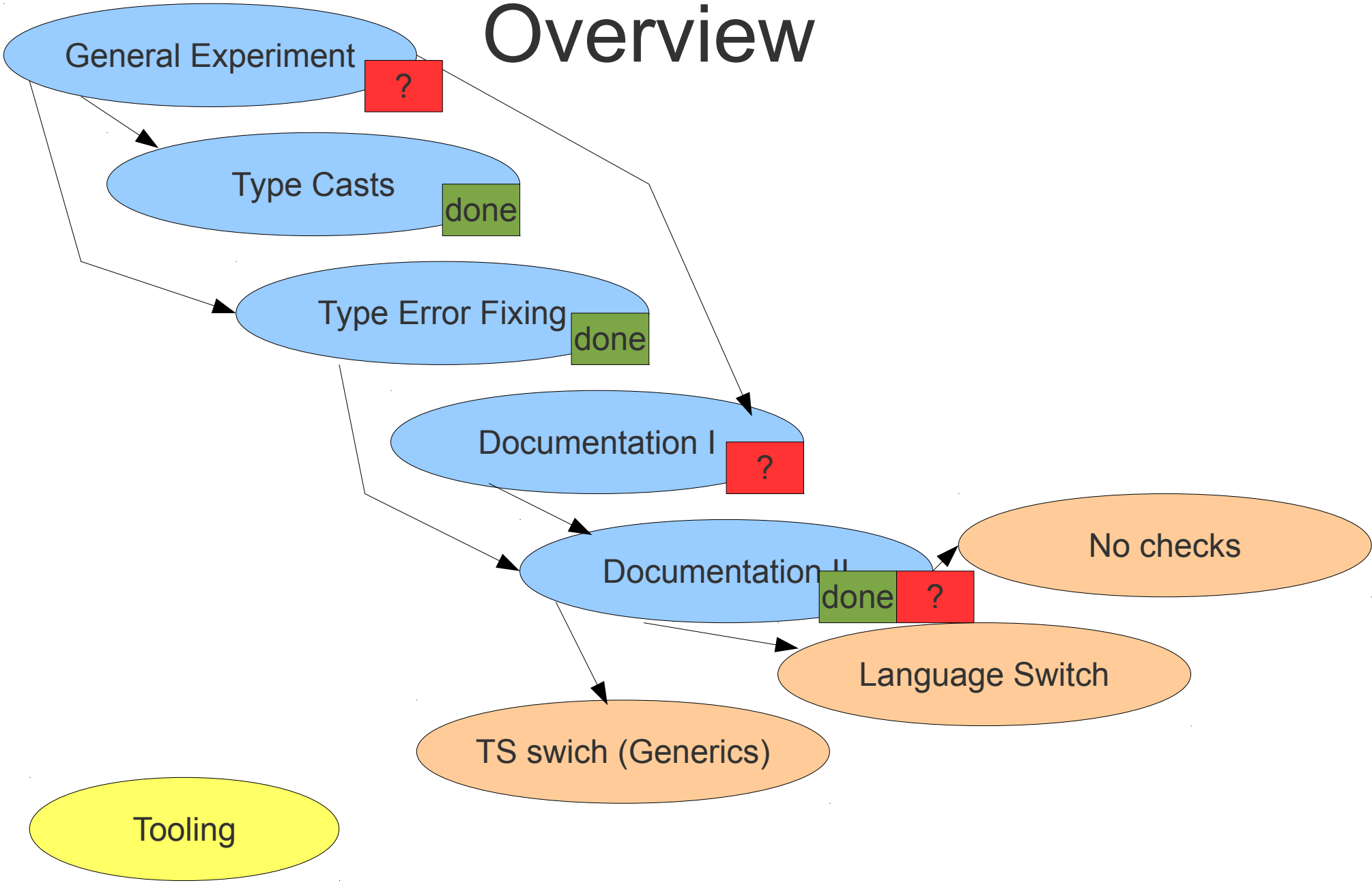
  speed of B** = **2 * speed of A

- **<u>A theory reveals how it can be tested</u>**

Hypothesis testing in Type Systems

- What is the theory of type systems?
  - large set of statements in literature....
    - *better program structures*, *help fixing bugs*, *better* documentation, etc.
  - **no clear theory that reveals how to test**
    - ***Example „better program structures“: how to measure?***

Overview



= experiments done	= in preparation / currently running	= in preparation

First Experiment (1) [OOPSLA'10]

- Idea
  - Experiment similar to Gannon'77, PrecheltTichy'98
  - Measure number of errors / time to completion
  - Make programming task larger
    (more generalizable?)

- How
  - ~50 subjects write parser / scanner
  - ~40 hours / subject = **1000 subject hours**

- Results
  - Dynamically typed people faster with scanner, no difference in parser
  - Opposite to Gannon'77, PrecheltTichy'98

First Experiment (2) [OOPSLA'10]

- Interpretation
  - There is at least one situation where static TS was counter productive
  - Falsification of „run an experiment and see the benefit of static TS“

- Personal conclusion
  - Experiment much too expensive
  - Relatively few insights
  - Unclear what the additional insights are

- What's next?
  - Many alternatives...
  - Try to identify often mentioned statements in literature
    - Type casts are bad for programmers, Type error fixing time better with static TS

Second Experiment (1) [DLS'10]

- Idea
  - Test „type casts are bad“
  - Only time to completion as dependent variable
  - More tasks, smaller tasks
- How
  - ~21 subjects write very small programs (3-10 LOCs)
  - ~4 hours / subject = **85 subject hours**
- Results
  - For small tasks casts matter (decrease productivity)
  - For larger tasks (10 LOC) no difference measured

Second Experiment (2) [DLS'11]

- Interpretation
  - Casts are not relevant enough for further studies
- Personal conclusion
  - Small experiments work
  - The more measurements the better
  - Change in experimental design worked well
- What's next?
  - Go on with often mentioned statements in literature
    - Type error fixing time better with static TS

Third Experiment (1) [Unpublished'11]

- Idea
  - Measure time until type error is fixed
  - Time to completion as dependent variable
  - Again more tasks, smaller tasks
- How
  - ~30 subjects, **120 subjects hours**
- Results
  - **Clear** benefit in fixing time

Third Experiment (2) [Unpublished'11]

- Interpretation
  - Type error fixing time validated without doubt
  - No idea how often this situation occurs in programming (controlled experiments won't help here)

- Personal conclusion
  - Fixing time considered as stable knowledge
  - Go on with different experiment, check fixing time from now on from time to time

- What's next?
  - Go on with often mentioned statements in literature
    - Type annotations as documentation

4th Experiment (1) [OOPSLA'12]

- Idea
  - 5 programming tasks on ondocumented API (only source code)
  - Time to completion as dependent variable
- How
  - ~30 subjects, **210 subject hours**
- Results
  - No clear results, 3 tasks show benefit of static TS (with annotations), 2 benefit of dynamic types (!?!)

4th Experiment (2) [OOPSLA'12]

- Interpretation
  - Ups....no clear interpretation
  - What about „bad luck“?
- Personal conclusion
  - Try to build up experiment from scratch, re-run it
  - There are situations where TS seem to be counterproductive
- What's next?
  - Re-run experiment

5th Experiment (1) [ICPC'12]

- Idea
  - 9 programming tasks, 2 type error fixing tasks, (2 semantic errors fixing tasks), 5 documentation tasks
- How
  - ~30 subjects, **120 subjects hours**
- Results
  - Type Error fixing time confirmed, now clear results in documentation pro TS

The next experiments (6-10)

- 6th experiment: Java Generics (just finished)
  - 18 subjects
  - Documentation, type error fixing time, extensibility

    => type error fixing time now „unclear“?!?!
  - Interpretation
    - Go on with Generics

- 7th experiment: Type annotations without checks (currently running)
  - 18 subjects coding in Dart
  - Documentation with & without type errors

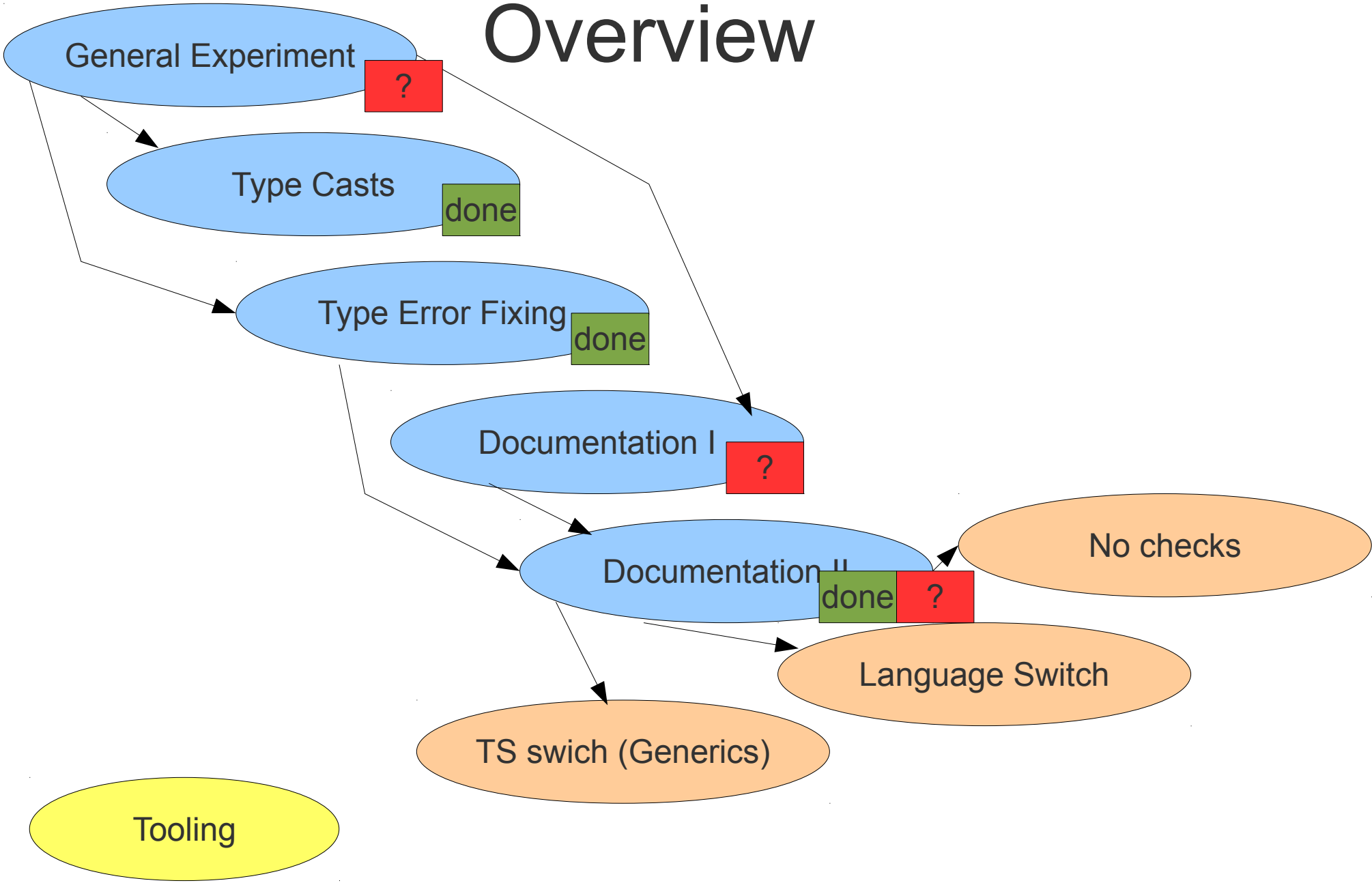
    => positive documentation impact without checks!

    => negatative documentation impact when types are wrong!

The next experiments (6-10)

- 8th experiment: Different languages (about to run)
  - Documentation tested
- 9th experiment: Documentation without annotation? (February '13)
  - Type inference
- 10th experiment: Documentation without annotation? (April'13)
  - Code completion

Overview



= experiments done	= in preparation / currently running	= in preparation

What's interesting in the series?

- Research method
  - Each experiment follows rigorous method
    - Hypothesis testing / 3-5 month pro experiment / small sample sizes
  - Do not generalize! (idealistic perspective)
  - Do generalize! (practical perspective)
    - Hardly experiment repetitions so far, but „assumed validity of results“
    - Following experiments massively influenced by previous ones
- So far
  - TS have positive impact on documentation, type error fixing time
  - Casts are no big deal
  - Generics are slightly „different“

Personal conclusion

- Go on measuring
  - Hopefully, this leads to a theory
- Follow rigorous methods
- Use small sample sizes
- Be aware that transitions between experiments are biased

Empirical Evaluation of static type systems: A running experiment series

Stefan Hanenberg
University of Duisburg-Essen, Germany

Austin, Texas, 07.12.2012