Teamscope[1]:

Measuring Software Engineering Processes with Teamwork Telemetry for Project Evaluation

An Ju (an_ju@berkeley.edu), Armando Fox

1. Team Evaluation via Automated Metric Summarization for Communication, Organization, Process, and Evolution

Team projects are good[1]

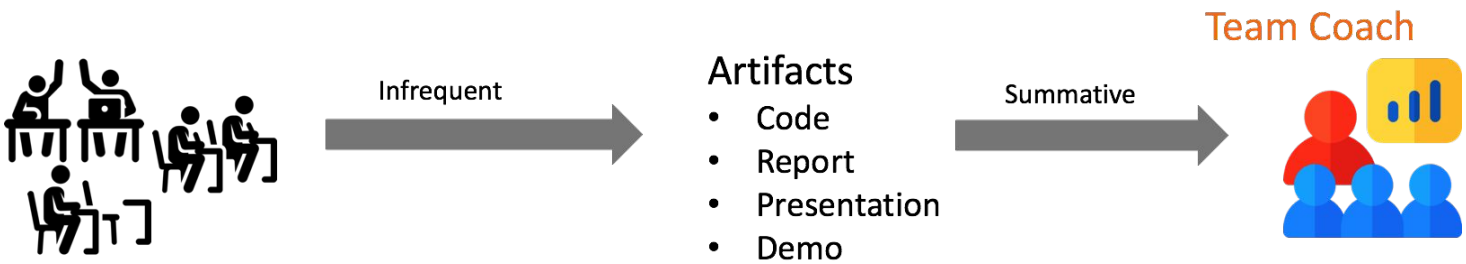
1. Krajcik, Joseph S., and Phyllis C. Blumenfeld. *Project-based learning*. na, 2006.
Fincher, Sally, and Marian Petre, eds. *Computer science education research*. CRC Press, 2004.

Team projects are good, but hard to scale.

Software engineering projects are hard to scale

Management[1]

Evaluation[2]



1. Devedzic, Vladan. "Teaching agile software development: A case study." IEEE Transactions on Education 54.2 (2011): 273-278.
2. Richards, Debbie. "Designing project-based courses with a focus on group formation and assessment." ACM Transactions on Computing Education (TOCE) 9.1 (2009): 2.

An opportunity from development tools

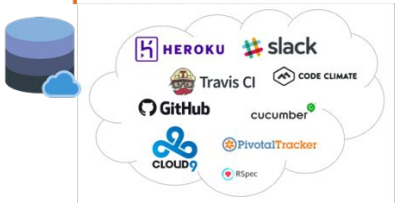
Infrequent

Artifacts
- Code
- Report
- Presentation
- Demo

Summative

Team Coach

Development Tools



Opportunity!

Software development and collaboration tools.

- GitHub: code hosting and collaboration
- Pivotal Tracker: task management
- Code Climate: code quality tracking
- Travis CI: integration
- ...

Data from development tools (teamwork telemetry) has information about both the software product and the team process that used to require intensive coach efforts to get.

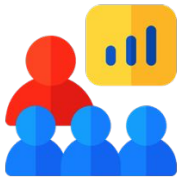
Measurements of Processes

Infrequent

Artifacts

- Code
- Report
- Presentation
- Demo

Summative

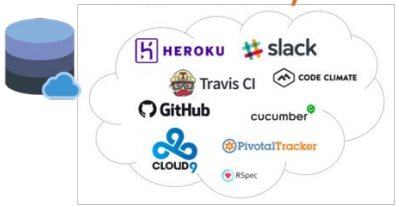


Real-time

Direct feedback

Formative

Teamwork Telemetry



Automated

- Processes directly measurable
- Metrics as proxy of process conformance

Measurement of Processes

- Code editing history
- Division of work
- Velocity
- Communication
- …

How Measurements of Processes are Designed

Workflow

- *Infer* team behaviors from teamwork telemetry
- *Detect* potential violations of best practices
- *Visualize* this info for coaches & teams
- *Determine* which process(es) most critical to success, & thresholds on soft violations

Based on concept of conformance templates[1]

1. N. Zazworka et al., “Are developers complying with the process: an XP study.” Proc. 2010 ACM-IEEE Intl. Symp. on Empirical Software Engrg. & Measurement (ESEM’10)

Hypotheses

H1. Following processes can promote expected outcomes.

H2. Metrics are good measurements of process conformance.

Methods

Case study

- A upper-level software engineering course at UC Berkeley
- Agile development method
- 123 students grouped into 21 teams
- Develop a web service in collaboration with customers
- Four 2-week iterations (sprints)

All data points (team, iteration) from a software engineering course

Classify data points into two groups based on metric scores (measured from teamwork telemetry).

Compare self-reported scores (measured from a self-assessment survey[1]) of two groups.

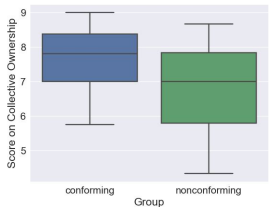
1. William Krebs. 2002. Turning the knobs: A coaching pattern for XP through agile metrics. Extreme Programming and Agile Methods-XP/Agile Universe 2002 (2002), 60–69.
Todd Sedano, Paul Ralph, and Cécile Péraire. 2017. Software development waste. In Proceedings of the 39th International Conference on Software Engineering. IEEE Press, 130–140.

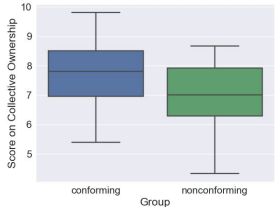
Results

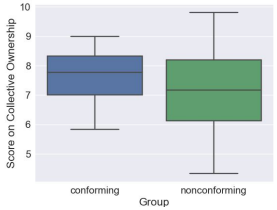
Outcomes ← H1 ← Processes ← H2 ← Metrics

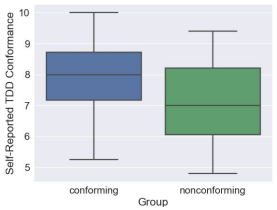
H1. Following processes can promote expected outcomes. ✓

H2. Metrics are good measurements of process conformance. ✓









Three processes promoting *collective ownership* are selected. ($p<0.05$ for first two figures)

A metric for Test-Driven Development is designed with GitHub data. ($p<0.05$)

Future work

More understandings on processes in software engineering project

Metrics on interaction and communication

An automated system as platform for coaching and managing software engineering teams

</talk>