A Software Engineering Capstone Infrastructure that Encourages Spreading Work Over Time and Team

***Abstract*—How can instructors facilitate spreading out the work in a software engineering or computer science capstone course across time and between team members? Currently teams often compromise the quality of their learning experience by frantically working before each deliverable. Some team members further compromise learning by not contributing their fair share to the team effort. To mitigate these problems, we propose using a GitHub template that contains all the initial infrastructure a team needs, including the folder structure, text-based template documents and template issues. In addition, we propose each team begins the year by identifying specific quantifiable individual productivity metrics for monitoring, such as the count of meetings attended, issues closed and number of commits. Initial data suggests that these steps have an impact. In 2022/23 we observed 50% of commits happening within 3 days of due dates. After partially introducing the above ideas in 2023/24, this number improved to 37%. We experiment with different measures of fairness of workload distribution, including the ratio of maximum to minimum commits on a team, Jain's fairness index, and our own fairness index based on disparity between number of commits. Going forward we propose an experiment where commit data and interview data is compared between teams that use the proposed interventions and those that do not.**

***Index Terms*—software engineering; capstone; template repository**

I. Introduction

The workload for a software engineering or computer science capstone team project is often unevenly distributed over time and between team members. Teams typically work in frantic bursts of activity right before a deadline and then cease almost all activity until their next deadline. These work habits compromise the learning objectives of the course because the students do not have time to properly plan their activities or reflect on their work. The uneven distribution of effort between team mates is also problematic. Some students take on an unfair share of the work, causing them stress and possibly hurting their experience in other courses, while those investing less effort miss important learning opportunities. How can instructors mitigate these problems?

To address the uneven distribution of work, need to first think about why the problems exist. Not the same as the workplace. Other pressures on students. Not sure of expectations. Not sure where to begin. Peer pressure and social interactions that make it challenging to take charge of the group, or criticize other group members. [There must be some literature that talks about the challenges for student teamwork, teamwork in SE, teamwork for capstone projects, teamwork for SE capstone projects]

Ideas on what we can do about it at an abstract level - the forces we can use to direct students. We have grades and we have structure of the course and expectations.

Overview of ideas.

Roadmap of paper.

Making sure that BibTex is working [1].

II. Literature Review

May not need this if the literature is covered in the introduction.

III. Proposed Infrastructure

roadmap blurb

A. Structure and Timeline

Context of course. Double blind. figure showing the V model and the expected deliverables. The ideas in this paper could work for other structures, but this is the one adopted.

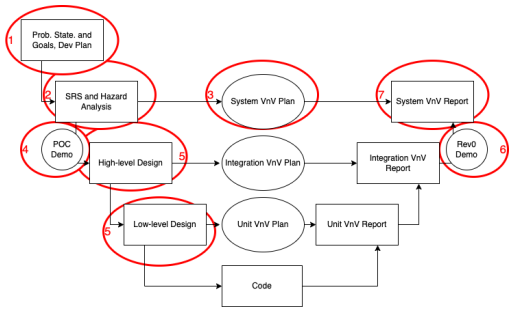

Fig. 1: V Model Used for Capstone Deliverables

B. Template Repository

link

C. Team Contribution Measurement

IV. Preliminary Data

Look at commits over time, and possibly lines (removing outliers) of code over time

A. Timeline Comparison

Timeline comparison

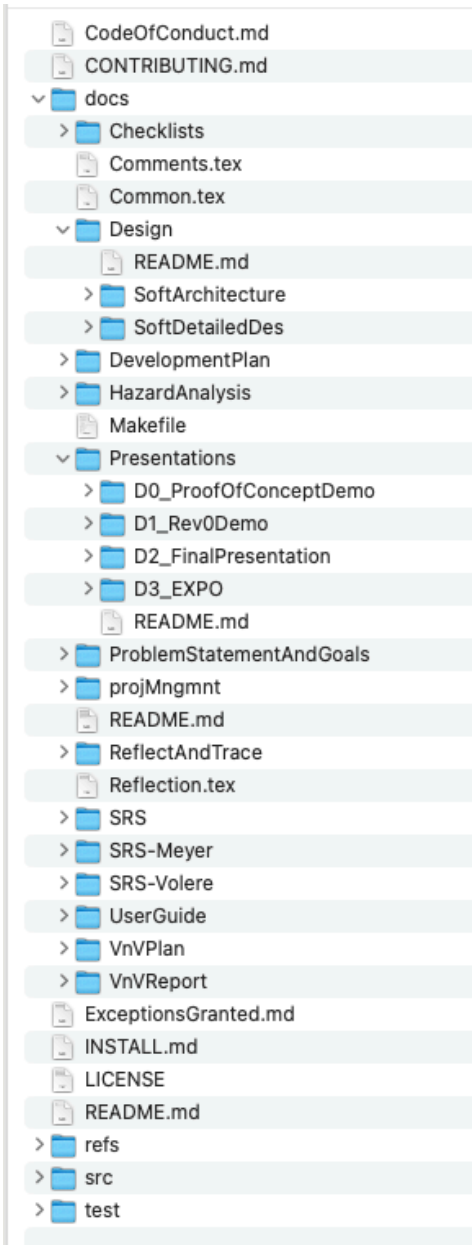

Fig. 2: GitHub Template

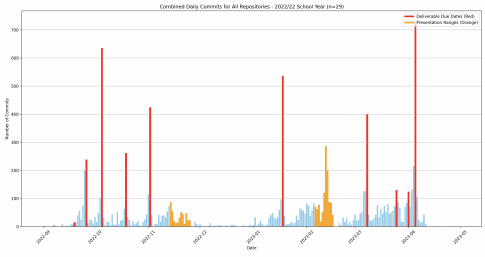

Fig. 3: Timeline of Commits for 2022–2023

B. Measuring Fairness

New fairness metric.

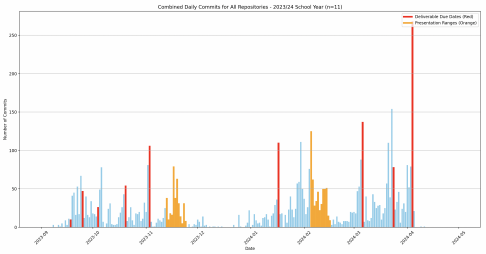

Fig. 4: Timeline of Commits for 2023–2024

$$\sum_{c,x \in C} (c > x \Rightarrow c - x)/((|C| - 1) \cdot \sum_{c \in C} c)$$

V. Proposed Experiment

Start with research questions.

Collect the same data as in Section IV and conduct focus groups in all three CAS capstone courses (SE, CS and TRON).

Threats to validity:

- Multiple changes are made to the course, so it is difficult to determine which change influences the student behaviour. The focus group should hopefully tease that out.
- Comparing different courses with different instructors, different backgrounds for students, etc.
- etc.

VI. Concluding Remarks

Acknowledgements

If any.

References

[1] W. S. Smith, "Teaching capstone software design project courses: Issues and challenges," in *Proceedings of the CASCON Workshops, Second International Workshop on Software Engineering Course Projects (SECP 2005)*, Markham, Ontario, 2005. [Online]. Available: https://www-927.ibm.com/ibm/cas/cascon/index.shtml (last accesed November 28, 2005)