Project Planning

Code Critiquer System for the C Language and Embedded C

sdmay25-23:
James Joseph
Samuel Lickteig
Alix Noble
Andrew Sand
Owen Sauser

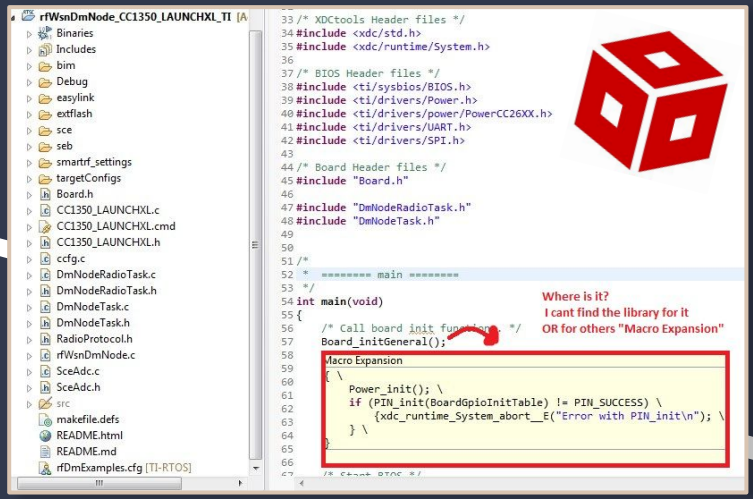
Code Critiquer System for the C Language and Embedded C

Project Overview

- Current state of project (continuation of sdmay24-34) is a web-based critiquer tool
  - Students upload C files to tool
  - Files are statically analyzed to search for antipatterns
  - Tool generates feedback
  - Students use feedback to improve skills
- Will modify current system and/or develop new prototypes
- Ideally tailored for CPR E 288
- Targeting a Spring Semester Prototype

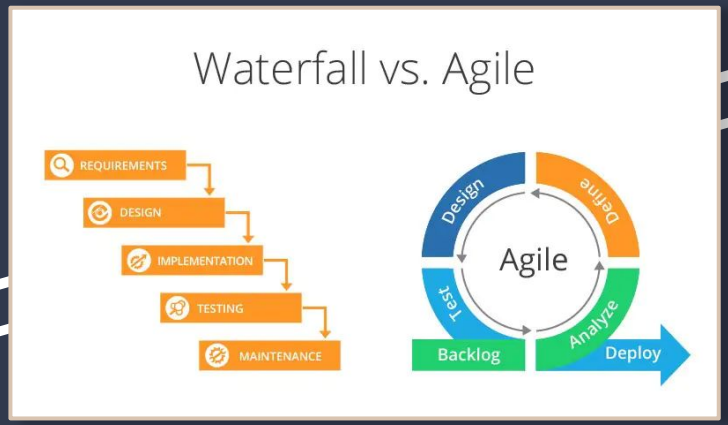
Problem Statement

- Static Code Analysis is a challenging problem
- Many off-the-shelf solutions
  - Many leave a lot to be desired
  - They are not bespoke for CPR E 288 usage
  - No off-the-shelf “perfect combination” for what the project client needs
- Client needs a code critiquer that can...
  - Be accessed by students and instructors
  - Provide beginner-oriented feedback
  - Ability to give embedded and datasheet-focused feedback
  - Potentially integrate with Code Composer Studio



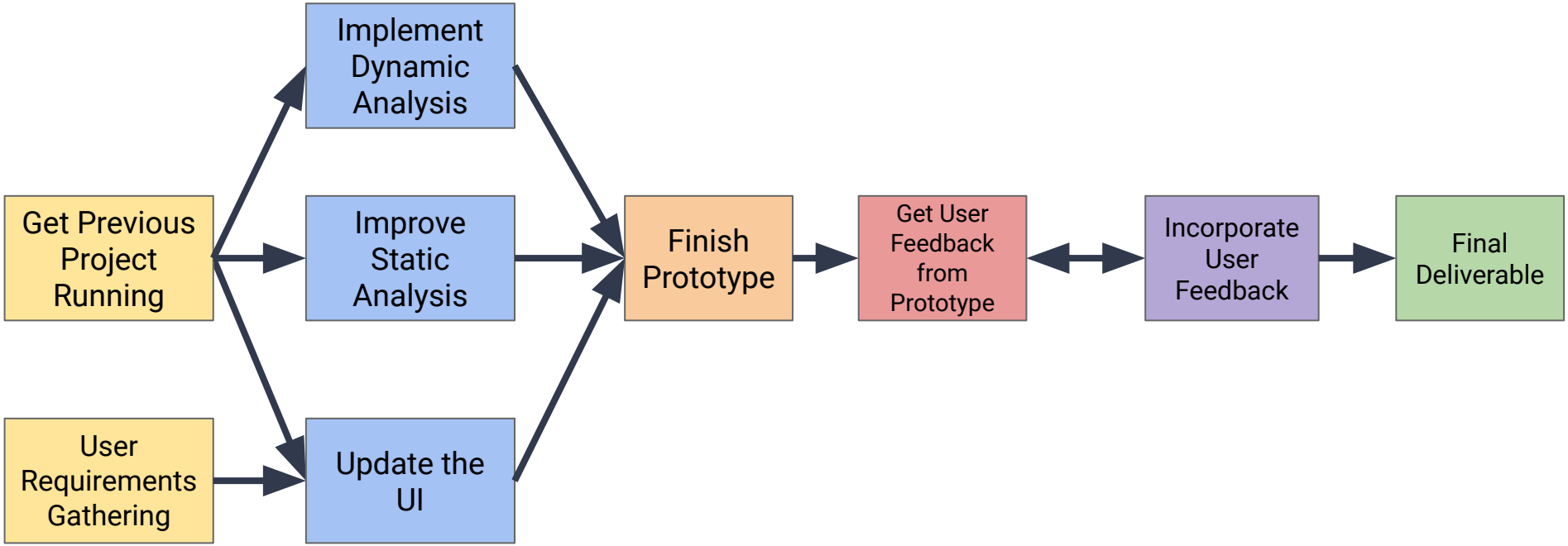
Project Management Style

Hybrid



- Hybrid - Drawing Inspiration from both Agile and Waterfall
  - Overarching project is broken down into Waterfall-like tasks
  - Each task is handled in an Agile “Sprint” manner
- Agile Elements:
  - Weekly Team and Advisor Meetings
    - Share Progress
    - Plan next “Sprint” (Week)
- Waterfall Elements:
  - Large tasks must be completed before project can advance (See Task Decomposition)

Task Decomposition (High Level)



Key Milestones, Metrics, and Evaluation Criteria

- Milestones
  - One antipattern per lab
  - Integration with Code Composer Studio
  - Integration with virtualized CyBot
- Metrics
  - Survey: negative or positive impact
  - Query professor, students, and TAs
- Evaluation Criteria
  - Positive impact on code quality
  - Decrease in TA debugging
  - Marginal impact on added effort

Key Risks

- We must ensure this tool is used for learning and not for cheating
- It is designed to help the students understand their errors, but not designed to fix the errors for them
- Correctness: The program should not mislead students and negatively affect their understanding of the C programming language
    - Need to be aware of “false positives”

Risk Mitigation Strategies

- We must ensure that the instructor/TA accounts are secure so that unfair homework information can not be obtained or malicious activities be executed
- The output given to the students must be informative and correct while also remaining within the scope of the assignment and not giving the students the answers outright

Conclusions from Project Planning Definition

- Most tasks are reliant on previous ones
- Impact is based on subjective metrics
- Feature development will start late due to an initial clean up and RE effort
- End product may evolve based on further findings during development
- More user research with a prototype will help focus development

Any Questions, Suggestions, or Comments?