CS 2820 Sprint 2 Presentation

Slide 1: Title Slide

CS 2820 Sprint 2 Presentation

Team Contributions and Plans for Sprint 3

Slide 2: Project Overview

Project Overview

- Focuses on analyzing and visualizing the reliability of end-to-end message transmissions in the WARP system.
- **Objective**: Significant progress on the `ReliabilityVisualization` class, ensuring integration with `-gui` and `-ra` options.

Slide 3: Sprint 2 Deliverables

Sprint 2 Deliverables

Code Updates:

- Implemented methods in `ReliabilityVisualization` and `VisualizationImplementation` for data creation, file output, and GUI visualization.
- Added high-level helper methods with JavaDoc comments to guide future implementation.
- Ensured compatibility with the `-gui` option, leveraging `ProgramVisualization` logic.
- Began implementing `createVisualizationData()` for reliability calculations.

UML Diagrams:

- Updated the sequence diagram to reflect new and modified methods.
- Adjusted the class diagram to include newly added or stubbed-out methods and their visibility (public/protected).

JUnit Tests:

- Created unit tests for all public/protected methods in `ReliabilityVisualization`.

Documentation:

- Added JavaDoc comments for all newly created or updated methods.
- Updated project documentation files with instructions for Sprint 2 progress.

Slide 4: Team Member Contributions

Team Member Contributions

Jeff Bates:

- Created JUnit tests to cover edge cases for all methods in `ReliabilityVisualization`.
- Tests were created for the following methods:
  - `ReliabilityVisualization.getReliabilities()`
  - `createVisualizationData()`
  - `createColumnHeader()`
  - `saveVisualization()`

Cooper Fort:

- Updated sequence diagram using `sequencediagram.org` to include new methods and saved it in the ARTIFACTS folder.
- Updated method calls in:
  - `ReliabilityVisualization.getReliabilities`
  - `WorkLoad.getFlowNamesInPriorityOrder`
  - `WorkLoad.getNodesInFlow`
  - `createHeader`
  - `createColumnHeader`
  - `createVisualizationData`
  - `saveVisualization`

Colin Miller:

- Refactored and cleaned up code for `ReliabilityVisualization`.
- Added logging for debugging in `createVisualizationData()` and `saveVisualization()`.

Yash Bandla:

- Updated the `README.md` with high-level plans for improving the `ReliabilityVisualization` class.
- Documented tasks and explained all artifacts.

Graham Besser:

- Created comprehensive JavaDoc comments for all methods in `ReliabilityVisualization`.
- Updated documentation files with method descriptions and parameters.

Slide 5: Artifacts

Artifacts

- **CS2820_Project ToDo**: Updated with Sprint 2 tasks and deliverables.
- **Sprint 2.pdf**: Detailed plans for task completion, UML diagram updates, and code changes.
- **WARP Sequence Diagram (V2)**: Reflects updated code flow for `ReliabilityVisualization`.

Slide 6: Plan for Sprint 3

Plan for Sprint 3

Objectives:

- Complete all unfinished methods in `ReliabilityVisualization` and `ReliabilityAnalysis`.
- Finalize JUnit tests for `ReliabilityAnalysis`.
- Implement error handling and ensure smooth user experience for both `-gui` and `-ra` options.
- Integrate all modules for end-to-end functionality.