Relatório de Melhorias

Engenharia de Software II

Prof. Dr. Marco Aurélio Graciotto
Ana Carolina Frozza

Considerando o PSP2.1, defina (nos mesmos moldes) um processo pessoal para desenvolvimento de software. Ao definir este processo, proponha uma melhoria, justificando-a adequadamente. Esta justificativa deve considerar o objetivo da melhoria, questões que permitam avaliar se o objetivo está sendo/foi alcançado e medidas que ajudem a responder as perguntas (elabore esta parte utilizando GQM).

No projeto do PSP2.1 foram analisados 8 fases de desenvolvimento.

- **Planejamento**: fase onde estima se o tempo e as funcionalidades a serem desenvolvidas no projeto, bem como a dificuldade e tamanho das mesmas;
- **Designer**: fase de modelamento das funcionalidades que foram criadas;
- **Revisão de Designer**;
- **Código**: fase de implementação das funcionalidades que foram criadas e modeladas;
- **Revisão de Código**: fase para analisar as funcionalidades no código;
- **Compilação**: fase para tornar o código executável;
- **Testes**: fase de validação das funcionalidades que foram implementadas;
- **Postmortem**: fase para relatar o tempo real gasto, o tamanho real das funcionalidades e se houve reutilização de código.

A **Tabela 1** é apresentado a estimativa de tempo e o tempo real gasto no projeto PSP2.1 após passar por todas as fases.

Tabela 1: Estimativa de Tempo

Projeto	**Tempo Estimado**	**Tamanho Estimado**	**Tempo Real**	**Tamanho Real**
PSP2.1	2:00	253	2:10	151

Podemos notar que a não há uma diferença muito grande entre o tempo estimado e o tempo real gasto no projeto e o tamanho real do projeto ficou menor do que o estimado, sendo assim podemos dizer que na fase de Planejamento não é necessário melhorias.
Visto que a fase de Designer e a fase de Revisão de Designer não apresentaram erros, podemos concluir que essas fases não necessitam de melhorias.
Considerando todos os projetos desenvolvidos na disciplina podemos obter uma melhor perspectiva em relação aos erros inseridos durante as fases.

Sendo assim, podemos observar no **Gráfico 1** que a grande maioria do erros foram inseridos no fase de Código e de Compilação, portanto podemos afirmar que essas fases precisam de melhorias.

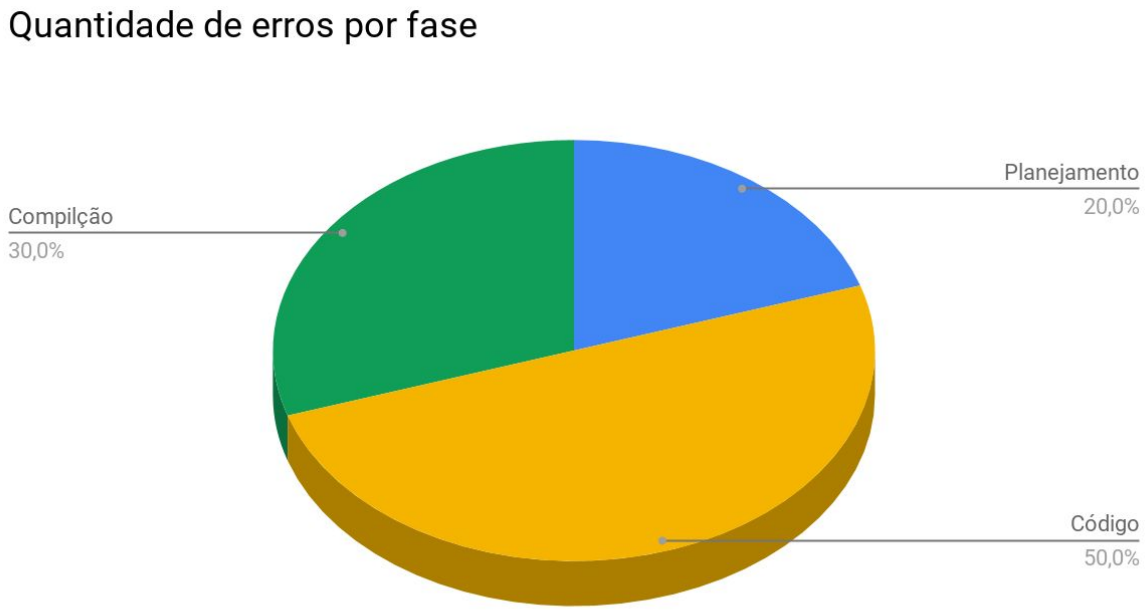

Gráfico 1: Erros por fase

Analisando os tipos de erros inseridos podemos notar que 60% desses erros foram do tipo Syntax, como apresenta o **Gráfico 2**.

Gráfico 2: Tipos de erros

Gráfico 3: Erros removidos por fase

Para melhoria da fase de Código visando os tipos de erros inseridos, é recomendado que o desenvolvedor análise com mais cuidado e calma o projeto antes de fazer o código e para melhoria tanto da fase de Código quanto para a fase de Compilação analisando os dados do **Gráfico 3** podemos sugerir que a fase de Código, revisão de Código, Compilação e Teste sejam feitas em uma única fase, considerando que para cada nova funcionalidade inserida no código fosse feita a compilação e o teste da mesma, poderíamos encontrar os erros com mais facilidade e impedir que esses erros se repitam por todo o código sobrecarregando assim a fase de Teste, visto que um erro inserido na fase de Código só é removido na fase de Teste (com base as porcentagens dos gráficos **1** e **2**).
Com a nova proposta das fases, seria gasto menos tempo de desenvolvimento pois ao invés de termos 8 fases teríamos 4.

- Planejamento;
- Designer;
- Revisão de Designer;
- Código, Revisão de Código, Compilação e Teste.

Assim a cada erro que fosse inserido já seria descoberto e resolvido na mesma fase, evitando que o desenvolvedor se perca na linha de raciocínio e consiga resolver o problema em menos tempo.

Como proposto o modelo GQM (Goal Question Metric) é apresentado pelo diagrama da **Figura 1,** o modelo apresenta um objetivo, as questões para se chegar a esse objetivo e as métricas que seriam as respostas para essas questão.

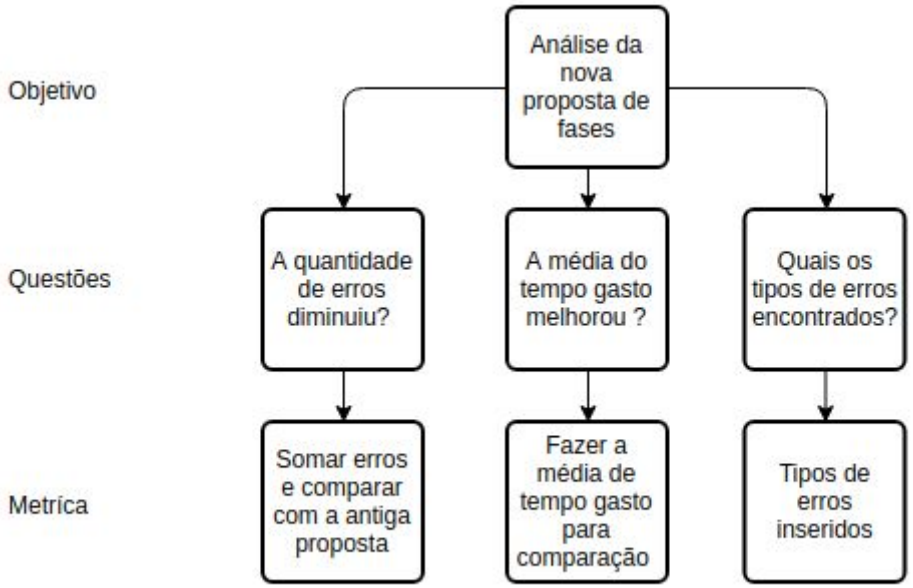

Figura 1: Diagrama GQM