Complexity Metrics

Cognitive Complexity (CogC)

Consideração de vários fatores, como declarações, loops, entre outros, avaliando o quão difícil é para um ser humano entender um trecho de código.

Valores elevados sugerem que um método pode ser difícil de entender (gráfico demonstra que seria fácil de entender).

Uma imagem com texto, captura de ecrã, ecrã, Retângulo

Descrição gerada automaticamente

Essential Cyclomatic Complexity (ev(G))

Baseada em estruturas de fluxo de controle (declarações, loops e saltos), medindo o nº de caminhos linearmente independentes por método.

Valores elevados indicam que pode ser mais difícil de testar (gráfico demonstra que seria fácil de testar).

Uma imagem com texto, captura de ecrã, Gráfico, Retângulo

Descrição gerada automaticamente

Design Complexity (iv(G))

Considera os dados que são lidos, modificados e usados ao longo dos métodos, medindo a complexidade dos fluxos de um método (chamadas de outros métodos).

Valores elevados podem indicar que o método está realizando muitas manipulações de dados, podendo tornando-o difícil de entender (gráfico demonstra que seria fácil de entender).

Uma imagem com texto, captura de ecrã, Gráfico, Retângulo

Descrição gerada automaticamente

Cyclomatic Complexity (v(G))

Calcula a complexidade cíclica de non-abstract methods, medindo o nº de caminhos executados por cada método. Também considera o nº mínimo de testes necessários para exercitar completamente o fluxo de controle de um método, isto é:

1 + (nº de: if’s, while's, for's, do's, switch cases, catches, expressões condicionais, &&'s e ||'s no método)

(gráfico demonstra que os métodos teriam uma baixa complexidade ciclica).

Uma imagem com texto, captura de ecrã, Retângulo, número

Descrição gerada automaticamente

Em resumo, o código possui algumas métricas de complexidade muito elevadas, o que indica que pode ser difícil de testar e entender. CogC , ev(G), iv(G)) e v(G) geralmente são indicadores de código que pode necessitar de refatoração e simplificação para melhorar a legibilidade, a manutenibilidade e a qualidade geral do código. Reduzir a complexidade e dividir métodos grandes em métodos menores e mais focados pode ajudar a tornar a base de código mais gerenciável e menos sujeita a erros.

A aplicação de design patterns podem ajuda a melhorar a organização e a estrutura do código.