Edeyson A. Gomes

Understanding Software Evolution using a Combination of Software Visualization and Software Metrics

Resumo

O artigo propõe uma solução que minimize a complexidade em analisar grandes volumes de dados, apontado como um dos maiores problemas ao se estudar a evolução de software.

A abordagem proposta centra-se na combinação de técnicas de visualização e métricas de software numa nova técnica denominada *evolution matrix* (matriz de evolução).

O objetivo da matriz de evolução é apresentar o comportamento de classes durante o ciclo de vida do sistema através de um vocabulário baseado em astronomia. Tal matriz deve suportar assertivas sobre a evolução do sistema com foco nas seguintes características: seu tamanho (em termo de classes), dinâmica de adição, remoção e estabilização de classes, tempo de vida de classes etc.

A visualização de classes proposta baseia-se em métricas como número de métodos e de atributos. Com base nas diferentes versões de uma classe classificam-na como pulsar, supernova, anã branca, gigante vermelha etc. A semântica empregada da astronomia indica o comportamento da classe, como o pulsar que incrementa e decrementa seu tamanho continuamente.

Crítica

O artigo apresenta claramente seu objetivo e usa uma metáfora deveras interessante para classificar o comportamento de classes durante a evolução do software. Porém, su primeiro estudo de caso é um sistema pouco tradicional, desenvolvido por um único indivíduo e com 38 versões.

Não fica claro se a análise de sistemas reais, onde as *releases* distem significativamente em tempo, serão tão claras quanto as apresentadas. Tampouco, se a expansão vertical da visualização degradará a percepção com um conjunto expressivo de refatorações de classes e pacotes.

Outro fator que deve ser mais explorado é o que causa os comportamentos usados na classificação do autor. Talvez o uso de classes utilitárias durante o desenvolvimento, como auxílio temporário à exploração de idéias de implementação, não devam nem sejam captadas entre *releases* do software.

A idéia é muito boa como semente para novas explorações na área e para testes com sistemas reais e maiores.

Questões para discussão

1. Classes utilitárias, que auxiliam a escolha da melhor implementação do código, devem fazer parte da classificação descrita?
2. Se sim, como representá-las entre as *releases* do software?
3. Que outras métricas poderiam ser utilizadas, combinadas às propostas do autor?
4. Que outras metáforas visuais poderiam ser utilizadas, combinadas à matriz de evolução?
5. Como solucionar o problema de visualização vertical para sistemas grandes?

Referência

Michele Lanza, Stéphane Ducasse, Understanding Software Evolution using a Combination of Software Visualization and Software Metrics. LMO 2002, pp. 135-149.