1. Introdução

Cada vez mais, os programas de computadores vem se tornando essencial para nós. Com isso, precisamos encontrar formas de criar e manter os códigos fonte da melhor maneira e considerando também a escalabilidade. Estudos, testes e outras formas de pesquisa foram surgindo para tratar esse tema. Foram criados padrões, termos e até algumas maneiras de encontrar vícios comuns entre o desenvolvimento de código para poder facilitar a resolução e agilizar o processo de atualização do software.

Com isso em mente, formas de refatorar código de maneira eficaz, rápida e sem muitos efeitos colaterais são os pontos mais desejados em uma evolução de *software*. Os efeitos colaterais acontecem quando, após uma edição em uma determinada etapa de refatoração, o código não se comporta como deveria em todos os pontos onde a parte alterada é referenciada, levando assim a um possível cascateamento de erros, quebrando todo o sistema. Todo esse processo é bem sensível a mudanças, principalmente em sistemas considerado legado. Por isso, a evolução dos métodos de refatoração segura e inteligente são cada vez mais necessários, para consigamos manter uma coerência dos resultados igual ao de antes da alteração. Outro problema que pode fazer com que a evolução de um *software* seja comprometida é a falta de atenção que o desenvolvedor possa ter na hora de editar e alterar os códigos. Por isso, a necessidade de tentar deixar alguns erros comuns já documentados.

A importância de se achar uma forma rápida, prática e barata de se fazer as refatorações necessárias sem efeitos colaterais e possíveis erros é gigantesca, tanto pelo mercado como pela evolução de sistemas científicos. Cada vez mais a depêndencia dos sistemas vem aumentando a críticidade das mudanças e, se for possível fazê-las sem que afetem sistemas essênciais, como sistemas elétricos; sistemas hospitalares, aviões e muitos outros que ajudam a melhorar a vida de cada vez mais pessoas.

Esse trabalho tem como foco principal, conseguir encontrar ferramentas e processos de refatoração de código para que a evolução de um software não cause problemas em partes já estabilizadas. Visa também, ajudar o leitor a conseguir evoluir suas habilidades de desenvolvimento e pensamentos lógicos ao montar o código. Quando se entende como a refatoração funciona, ao se escrever um novo trecho, esse formato já estará sendo levado em conta, deixando mais simples se houver uma necessidade futura de alteração.

O resultado esperado é que o método criado para refatoração mostre uma ajuda considerável na refatoração do código e com isso, uma facilidade para poder refatorar e evoluir o código.

Este artigo se estrutura da seguinte forma: na Secão 2 são apresentados os conceitos relevantes ao estudo. A Secção 3 contempla os trabalhos relacionados à pesquisa proposta. Na Seção 4, sãao apresentados os materiais e métodos utilizados para a execução do trabalho. A Seção 5 detalha os resultados obtidos a partir da mineração dos dados. Na Seção 6, são apresentadas as discussões dos resultados obtidos. Por fim, as Seções 7 e 8, apresentam as ameac¸as a validade e as conclusões deste trabalho, respectivamente.