3
Medição de Software

À medida que a engenharia de software amadurece, a medição de software passa a desempenhar um papel cada vez mais importante no entendimento e controle das práticas e produtos do desenvolvimento de software [18]. Alguns desenvolvedores medem características do software para ter alguma noção se os requisitos estão consistentes e completos, se o projeto tem boa qualidade ou se o código está pronto para ser testado. Alguns gerentes de projetos medem atributos do processo e do produto para serem capazes de dizer quando o software estará pronto para entrega ou se o orçamento será ultrapassado. Pessoas responsáveis pela manutenção podem usar a medição para avaliar se o produto precisa ser melhorado.

Em termos gerais, medição é o processo pelo qual números ou símbolos são designados a atributos de entidades do mundo real de forma a descrevê-los de acordo com regras claramente definidas [19]. Portanto, a medição captura informações sobre atributos de entidades. Uma entidade é um objeto (como uma pessoa ou um quarto) ou um evento (como uma viagem ou o projeto de desenvolvimento de um software). Um atributo é uma característica ou propriedade de uma entidade. Exemplos de atributos são a área de um quarto, o tempo de uma viagem ou o custo de um projeto de desenvolvimento de um software.

A primeira tarefa de qualquer atividade de medição é identificar as entidades e atributos que se deseja medir [18]. Na engenharia de software, existem três classes de entidades: processos, produtos e recursos [18]. Processos são coleções de atividades relacionadas ao desenvolvimento de software. Produtos são quaisquer artefatos que resultam de uma atividade do processo. Recursos são entidades requeridas para realizar uma atividade do processo.

Dentro de cada classe de entidade, é possível distinguir atributos internos e atributos externos [19]. Atributos internos de um processo, produto ou recurso são aqueles atributos que podem ser medidos ao se examinar diretamente o produto,

processo ou recurso separadamente do seu comportamento. Atributos externos são aqueles que só podem ser medidos em termos de como o processo, produto ou recurso se relaciona como seu ambiente. Neste caso, o comportamento é importante, e não apenas a entidade em si.

Para entender melhor a diferença entre atributos internos e atributos externos, considere um conjunto de módulos de um sistema de software. Sem executar o código, é possível determinar muitos atributos internos importantes: o tamanho (talvez em termos de linhas de código), sua complexidade (talvez em termos do número de pontos de decisão do código) ou a dependência entre os módulos. No entanto, os atributos externos só podem ser medidos quando o código é executado: por exemplo, o número de falhas encontradas pelo usuário, a dificuldade de navegação entre as telas ou a quantidade de tempo necessária para procurar uma informação no banco de dados.

Este trabalho de mestrado se preocupou em avaliar atributos de produtos de software. Produtos não são restritos apenas aos itens que são entregues ao cliente. Qualquer artefato produzido durante o ciclo de vida do software pode ser medido e avaliado. Existem muitos atributos externos de produtos de software. Confiabilidade, facilidade de compreensão, manutenibilidade, usabilidade, integridade, eficiência, reusabilidade, portabilidade e interoperabilidade são exemplos de atributos externos de produtos. Esses atributos não estão relacionados apenas ao código, mas também a outros documentos e artefatos que dão apoio ao desenvolvimento de software. Por exemplo, a manutenibilidade e a reusabilidade das especificações de requisitos e do projeto de um software são tão importantes quanto a manutenibilidade e a reusabilidade do código. Como será visto no Capítulo 4, o foco deste trabalho é avaliar a manutenibilidade e reusabilidade do projeto e do código de software orientado a aspectos.

Existem também muitos atributos internos de produtos. Por exemplo, especificações podem ser avaliadas em termos de seu tamanho, grau de reutilização, redundância e correção sintática. O projeto detalhado e o código de um software podem ser avaliados por esses mesmo atributos e mais alguns outros como, por exemplo, acoplamento, coesão e estrutura de fluxo de controle e de dados. Os usuários muitas vezes consideram os atributos internos como pouco importantes, pois eles se interessam principalmente com a funcionalidade final, a qualidade e utilidade do software. Contudo, os atributos internos podem ser muito

úteis para sugerir o que provavelmente será encontrado ao se avaliar os atributos externos [19]. Por exemplo, Yourdon & Constantine [20] assumem que projetos de software que possuem módulos com baixo acoplamento entre si e alta coesão dão origem a códigos mais confiáveis e fáceis de manter. Neste trabalho, os atributos internos de tamanho, acoplamento, coesão e separação de *concerns* foram utilizados para avaliar os atributos externos de manutenibilidade e reusabilidade.

3.1. Modelos de Qualidade de Software

Numa perspectiva de medição, qualidade de software deve ser definida em termos de atributos de produtos de software que são de interesse do usuário. Usuários de sistemas de software geralmente querem ser capazes de medir e predizer atributos externos, uma vez que o comportamento do sistema os afeta diretamente. A confiabilidade, a usabilidade e a portabilidade, por exemplo, afetam decisões de compra de sistemas. No entanto, atributos externos são normalmente mais difíceis de medir do que atributos internos, e eles só podem ser medidos tardiamente no processo de desenvolvimento [19]. Por exemplo, confiabilidade só pode ser medida depois que o desenvolvimento estiver completo e o sistema pronto para uso. Portanto, alguns desenvolvedores de software procuram usar atributos internos para predizer atributos externos, pois têm necessidade de monitorar e controlar os produtos durante o desenvolvimento. Por exemplo, sabendo da relação entre alguns atributos internos do projeto e a propensão a falhas ou dificuldade de manutenção do software, os desenvolvedores querem ser capazes de identificar os módulos, ainda no estágio de projeto, cujo perfil, em termos de medidas de atributos internos, mostra que eles são prováveis candidatos a mais tarde apresentarem alguma falha ou dificuldade para manter.

No entanto, muitas vezes é difícil definir os atributos externos de uma forma mensurável na qual todo mundo concorde. Todos querem construir e comprar sistema de alta qualidade, mas nem sempre concordam com o significado de qualidade, o que gera dificuldades para medir qualidade de uma forma compreensível. Essa observação levou os engenheiros de software a desenvolver modelos de qualidade que definem atributos externos em termos de atributos

internos, uma vez que os atributos internos são mais concretos, bem definidos e fáceis de medir.

Os modelos de qualidade são geralmente construídos de uma forma parecida com uma árvore. Os ramos superiores ou mais à esquerda representam os atributos externos que se deseja avaliar, como confiabilidade e usabilidade. Em muitos modelos de qualidade já desenvolvidos, esses atributos são chamados de fatores de qualidade de alto nível. Cada atributo externo é composto por atributos internos, como complexidade e modularidade. Os atributos internos também são chamados de critérios de baixo nível. Como os atributos internos são mais fáceis de medir do que os atributos externos, métricas reais são propostas para eles. Os modelos de qualidade representados em forma de árvore descrevem os relacionamentos pertinentes entre os atributos externos e os atributos internos que os influenciam, portanto os atributos externos podem ser medidos em termos dos atributos internos pelos quais eles são influenciados [19]. McCall [21] e Boehm [22] foram os primeiros a descrever qualidade de software usando a abordagem do modelo de qualidade. A Figura 3 mostra o modelo de qualidade proposto por Boehm.

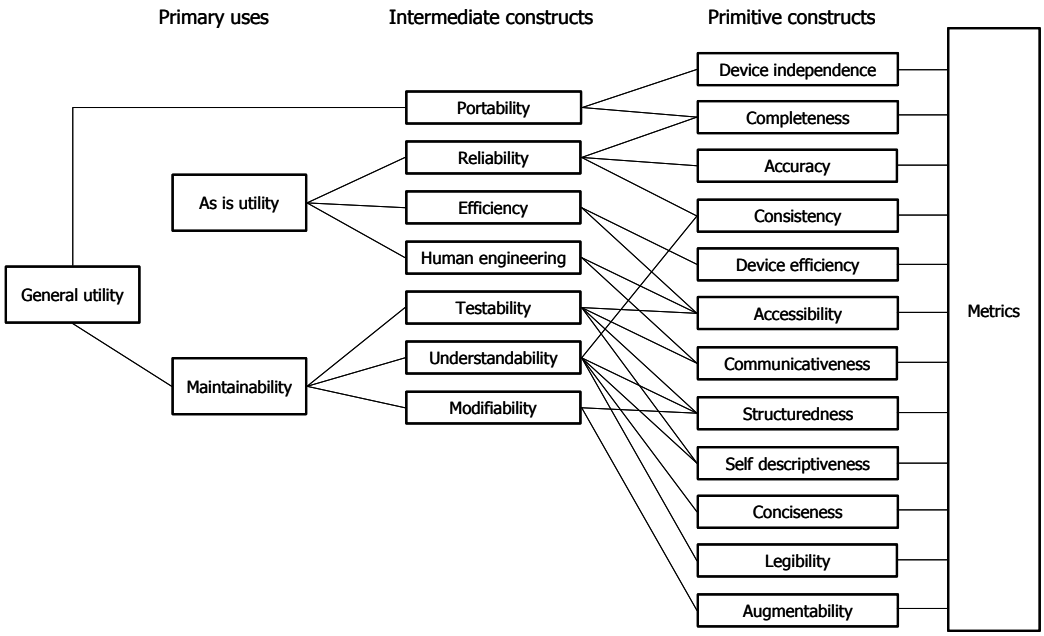

Figura 3 – Modelo de qualidade de Boehm

De acordo com [19], modelos de qualidade podem ser usados de duas maneiras:

- Um modelo já existente é escolhido e os relacionamentos entre os atributos externos, os atributos internos e as métricas são aceitos exatamente como propostos pelo autor do modelo.
- A filosofia geral de que um atributo externo é influenciado por vários atributos internos é aceita, mas não é adotado um modelo de qualidade já desenvolvido. Neste caso, um modelo de qualidade próprio é definido, baseado em modelos de qualidade e teorias já existentes.

Neste trabalho, a abordagem do modelo próprio foi utilizada. Um modelo de qualidade focado nos atributos externos de manutenibilidade e reusabilidade foi definido e será apresentado no Capítulo 4.

3.2. A Abordagem *Goal-Question-Metric*

Muitos processos de medição começam medindo o que é conveniente ou fácil de medir, em vez de medir o que é necessário. Normalmente, processos como estes falham, pois os dados resultantes não são úteis para quem desenvolve ou mantém o software [19]. De acordo com muitos estudos sobre a aplicação de métricas e modelos de qualidade em ambientes industriais, um processo de medição, para ser efetivo, tem que ser focado em objetivos específicos, e sua interpretação deve ser baseada na caracterização e no entendimento desses objetivos [23]. A abordagem *Goal-Question-Metric* (*GQM*) [23] se baseia nessa crença e foi definida originalmente por Basili para avaliar defeitos em uma série de projetos do Centro de Vôo Espacial da NASA. Depois foi aplicada em vários outros contextos.

O paradigma *GQM* é uma abordagem orientada a objetivos para a medição de produtos e processos de software, que apóia a definição *top-down* do processo de medição e a análise *bottom-up* dos dados resultantes. Ela tem uma série de vantagens: ajuda na identificação de métricas úteis e relevantes; apóia a análise e interpretação dos dados coletados; permite uma avaliação da validade das

conclusões tiradas; e diminui a resistência das pessoas contra processos de medição [24].

A abordagem *GQM* pode apoiar a medição de qualquer tipo de produto ou processo, cujo propósito seja qualquer um, desde a caracterização, até o controle e aperfeiçoamento, cujo foco seja qualquer atributo de qualidade, definido sob qualquer perspectiva e em qualquer ambiente [24]. A abordagem GQM provê um framework que envolve três passos:

1. Listar os principais objetivos do processo de medição;
2. Derivar de cada objetivo as perguntas que devem ser respondidas para determinar se os objetivos foram atingidos;
3. Decidir o que precisa ser medido para ser capaz de responder as perguntas adequadamente (definição das métricas).

Os objetivos da medição são definidos em termos da entidade, propósito, atributos de qualidade, ponto de vista e ambiente (por exemplo, analisar o **sistema de software** com o propósito de fazer **predição** da **confiabilidade** do ponto de vista do **cliente** na **empresa XYZ**). Cada objetivo é refinado em um conjunto de perguntas que representam uma definição operacional do objetivo (por exemplo, qual a distribuição de falhas por nível de gravidade?). Para cada pergunta, as métricas relevantes são definidas (por exemplo, número total de falhas e número de falhas gravíssimas). A figura 3 mostra a estrutura hierárquica gerada pela utilização da abordagem *GQM*. Neste trabalho a abordagem *GQM* foi usada para organizar o processo de medição e definir as métricas do framework de avaliação. O Capítulo 4 apresenta o objetivo e as perguntas geradas, enquanto o Capítulo 5 define as métricas do framework de avaliação.

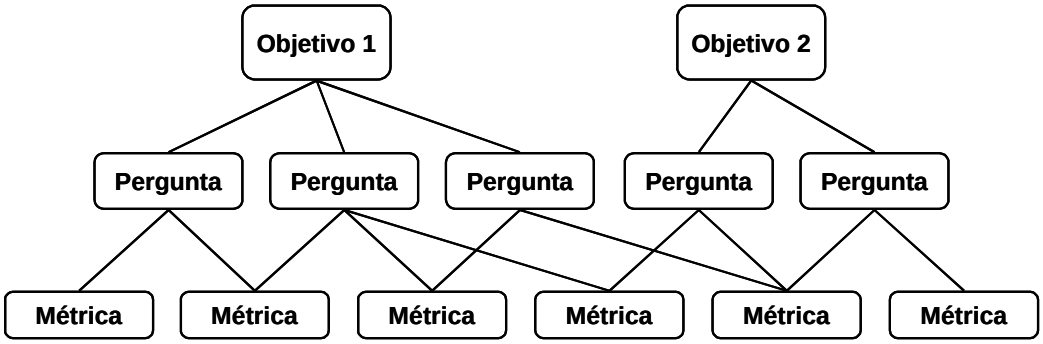

Figura 4 – Estrutura hierárquica da abordagem GQM