Transcrição Consolidada
MD395
Qualidade de Software
Prof. Washington Almeida

ATENÇÃO!

Esta é uma **transcrição consolidada** da fala do professor usando um algoritmo com alto nível de confiança. Porém, apesar disso, pode haver alguns poucos erros de reconhecimento de fala (geralmente de fácil percepção). O objetivo não é ser uma apostila sobre o assunto, mas somente uma transcrição. Na dúvida, assista às próprias aulas gravadas.

Importante: aulas de **resolução de questões** não foram transcritas, pois as questões em si já são uma revisão da mesma teoria.

Aula 01

Olá, pessoal. Vamos abordar hoje a temática da qualidade de software. Sou o Professor Washington Almeida, especializado em engenharia de software, e minhas pesquisas também se concentram nessa área, principalmente nas métricas relacionadas à qualidade. Nossa discussão será centrada no conceito de qualidade de software. Esse tópico engloba o gerenciamento da qualidade do software, que tem como objetivo assegurar que os sistemas desenvolvidos atendam aos seus propósitos. Isso envolve garantir que o sistema atenda às necessidades dos usuários, funcione eficientemente, cumpra requisitos não funcionais, verifique o processo de construção e seja entregue dentro do prazo e orçamento estabelecidos.

A gestão da qualidade de software abrange diversos aspectos. Além de satisfazer os usuários e funcionar eficientemente, é crucial que os projetos sejam concluídos conforme o cronograma e o orçamento definidos. Essa gestão está intrinsecamente ligada às características do projeto. Portanto, é necessário que o software seja não apenas funcional e eficiente, mas também desenvolvido no tempo previsto e sem exceder os recursos financeiros planejados.

No contexto da definição de qualidade de software, é relevante considerar os requisitos, especialmente os não funcionais. A distinção entre requisitos funcionais e não funcionais é fundamental para compreendermos a qualidade do software. Enquanto os requisitos funcionais descrevem as funcionalidades do sistema, os requisitos não funcionais, também conhecidos como critérios de qualidade, são cruciais para avaliar a qualidade do software. Essas divisões estabelecem as bases para compreender o conceito de qualidade em software.

A produção de software envolve métodos, processos, técnicas e padrões de desenvolvimento. A diferença entre garantia de qualidade e controle de qualidade é essencial para entender o processo. A garantia de qualidade diz respeito a funções gerenciais, como auditorias e relatórios. Isso implica acompanhar o processo de construção do software e garantir que os padrões estejam sendo seguidos. Por outro lado, o controle de qualidade envolve inspeções, revisões e elementos utilizados ao longo do processo. Esses conceitos foram adaptados da indústria e aplicados à engenharia de software, considerando sua natureza mais recente.

A engenharia de software tem a qualidade como um de seus principais focos. Para abordar essa questão, são empregados processos, métodos e ferramentas. A figura do Pressman representa as camadas da engenharia de software e destaca os desafios enfrentados em seus primórdios, como orçamento, prazos, requisitos não atendidos e dificuldades de manutenção. O objetivo central da qualidade de software é solucionar esses problemas, garantindo o cumprimento dos prazos e orçamentos, atendimento aos requisitos do usuário e eficiência operacional.

A evolução tecnológica e a aplicação de técnicas de gerenciamento de qualidade, juntamente com métodos de teste, resultaram em melhorias significativas na qualidade do software nas últimas décadas. Isso permitiu uma avaliação mais simples e eficaz dos elementos de qualidade do software, resultando em produtos finais de melhor desempenho. A estrutura de qualidade de software envolve características de qualidade, subcaracterísticas, medidas e métricas, que serão abordadas nas aulas subsequentes.

A relação entre garantia de qualidade e controle de qualidade provém de métodos empregados na indústria manufatureira. A garantia de qualidade diz respeito à definição de processos e padrões para garantir produtos de alta qualidade, enquanto o controle de

qualidade envolve a aplicação desses processos para avaliar e eliminar produtos que não atendem aos padrões de qualidade. Essa abordagem tem seu fundamento na indústria e se relaciona diretamente com os conceitos de qualidade de software.

A gestão formal da qualidade é especialmente crucial em projetos de software complexos e de longa duração, que exigem um planejamento minucioso e um controle rigoroso. Essa abordagem é aplicada em projetos que têm uma visão estruturada, seguindo planos definidos. Em projetos desse tipo, a equipe de gerenciamento da qualidade assume a responsabilidade de definir processos de desenvolvimento, padrões e métodos a serem aplicados, garantindo a qualidade tanto do processo quanto do produto final.

No âmbito organizacional, a qualidade do software é amplamente defendida, especialmente em projetos complexos que seguem abordagens baseadas em planos. Aqui, há uma preocupação constante em estabelecer processos e padrões para garantir produtos de alta qualidade. A equipe de gerenciamento da qualidade desempenha um papel central, definindo os processos a serem seguidos, os padrões a serem aplicados à documentação e as metas de qualidade do projeto.

Além disso, há um enfoque no gerenciamento da qualidade do projeto, que envolve a aplicação de processos específicos de qualidade e a verificação do cumprimento dos planos definidos. A qualidade do projeto é avaliada em cada etapa, garantindo que o resultado final esteja em conformidade com os padrões estabelecidos. Essa abordagem é especialmente relevante em projetos complexos, onde a organização é fundamental para garantir a qualidade do processo e do produto.

Em resumo, a qualidade de software é um aspecto crucial da engenharia de software. Ela abrange desde a definição de processos e padrões até a aplicação de medidas de controle e verificação da qualidade. Esses conceitos têm suas origens em métodos de garantia e controle de qualidade utilizados em outras indústrias, adaptados à realidade da engenharia de software. Essa abordagem abrange desde a definição dos requisitos até a entrega do produto final, com foco constante na satisfação do cliente e na eficiência operacional.

Aula 02

Vamos abordar a distinção entre garantia de qualidade e controle de qualidade dentro do contexto da engenharia de software. Tanto a garantia de qualidade quanto o controle de qualidade fazem parte do gerenciamento global da qualidade. Isso já ficou claro para você. No âmbito da indústria de software, diferentes empresas podem adotar abordagens variadas para a garantia de qualidade. Algumas encaram a garantia de qualidade como a definição de procedimentos, processos e padrões para assegurar a qualidade do software desde o início. Por outro lado, outras empresas entendem a garantia de qualidade como um conjunto de atividades que inclui, além da definição de processos, também a gestão de configuração, a verificação e a validação aplicadas após a entrega do produto pelo time de desenvolvimento. Portanto, essa dualidade de abordagens ilustra a amplitude da garantia de qualidade no contexto da indústria de software.

Dito isso, é importante ressaltar que a diferença entre controle e garantia de qualidade no setor de software é mais sutil do que em comparação a setores de manufatura. Em manufatura, a distinção é mais clara e tangível, pois um produto físico pode ser submetido a testes de controle de qualidade após sua produção, enquanto a garantia de qualidade se concentra em estabelecer os padrões e processos a serem seguidos. No entanto, no desenvolvimento de software, a natureza contínua do ciclo de vida do produto torna essa diferenciação menos nítida.

Para reforçar essa compreensão, vamos esmiuçar a diferença entre controle e garantia de qualidade no âmbito da engenharia de software. O controle de qualidade, em suma, ocorre ao longo do processo de desenvolvimento do software e concentra-se na verificação e validação contínuas para assegurar que o software esteja em conformidade com os padrões de qualidade estabelecidos. Esse processo envolve inspeções, revisões e testes que ocorrem durante todo o ciclo de vida do software. O controle de qualidade está intimamente relacionado ao processo de produção, com o objetivo de monitorar a qualidade durante sua construção.

Por outro lado, a garantia de qualidade, no contexto do software, amplia-se para além da fase de desenvolvimento. Ela se concentra em funções gerenciais que incluem a auditoria e o relato, buscando garantir que os processos e produtos atendam aos padrões estabelecidos mesmo após o lançamento do software. A garantia de qualidade está mais ligada a assegurar a qualidade do software em produção, verificando se ele atende às expectativas dos usuários finais e mantém os padrões de qualidade.

Uma abordagem interessante para compreender a diferença entre garantia e controle de qualidade no contexto do software é trazer uma definição clara de uma das atividades. A garantia de qualidade do software busca assegurar que os processos e produtos ao longo do ciclo de vida do projeto estejam em conformidade com os padrões, procedimentos e descrições de processos definidos para o projeto. Já o controle de qualidade, que se estende durante o desenvolvimento do software, envolve inspeções, revisões e testes para verificar se o software atende aos padrões definidos.

Com essas explicações, torna-se evidente que, na engenharia de software, as funções de controle e garantia de qualidade interagem e se complementam ao longo do ciclo de vida do produto. Embora não haja uma separação rígida entre essas atividades, é essencial entender suas diferenças sutis para garantir que o software atenda aos requisitos de qualidade esperados tanto durante a sua construção quanto após o seu lançamento.

Aula 03

Vamos agora explorar a perspectiva da qualidade nos processos com base em abordagens mais tradicionais e orientadas por planos. Isso é particularmente relevante em cenários onde há uma estabilidade maior nos requisitos e menos mudanças ao longo do desenvolvimento. Estamos abordando o contraste com o desenvolvimento ágil, mas, por enquanto, estamos concentrados na abordagem mais convencional.

Nesse contexto, a gestão da qualidade desempenha um papel crucial, oferecendo uma verificação independente do processo de desenvolvimento de software. Uma equipe de qualidade é responsável por avaliar os resultados do projeto para assegurar que estejam em conformidade com os padrões e objetivos organizacionais. Além disso, essa equipe verifica a documentação do processo, que registra as tarefas realizadas por cada equipe envolvida no projeto.

Em ambientes onde os requisitos são mais estáveis, essa equipe de qualidade desempenha um papel fundamental. Ela atua antes do software ser entregue aos clientes, examinando e avaliando o sistema para garantir que atenda aos critérios e requisitos estabelecidos. A equipe de qualidade também garante que os testes do sistema abordem adequadamente os requisitos e mantém registros precisos do processo de teste.

Essa equipe deve ser independente do grupo de desenvolvimento de software para manter uma visão objetiva da qualidade do produto. Essa independência permite que a equipe de qualidade reporte a qualidade do software sem ser influenciada por problemas ou pressões relacionados ao desenvolvimento. A importância dessa independência é destacada em cenários onde prazos e orçamentos podem impactar a qualidade do produto final.

Idealmente, o gerenciamento da qualidade deve ser responsável por toda a organização e deve reportar a um nível de gerenciamento acima do gerente de projeto. Isso assegura que a qualidade não seja comprometida por limitações de tempo ou recursos. Um plano de qualidade é fundamental nesse contexto. Ele define os atributos mais importantes de qualidade para o software sendo desenvolvido.

Para elaborar um plano de qualidade sólido, é necessário considerar cinco tópicos essenciais. Primeiro, uma introdução ao produto, incluindo uma descrição do produto, o mercado-alvo e as expectativas de qualidade. Em segundo lugar, os planos do produto, com datas críticas de lançamento, responsabilidades e planos de distribuição e manutenção. O terceiro tópico aborda a descrição dos processos e padrões de desenvolvimento e serviço utilizados.

O quarto ponto destaca as metas de qualidade, que incluem os planos para garantir a qualidade do produto, identificação de atributos críticos e gerenciamento de riscos. Finalmente, um mapa de riscos é fundamental, onde os principais riscos que podem afetar a qualidade do produto são identificados, juntamente com as ações necessárias para mitigá-los.

Vale destacar que a qualidade do software está diretamente relacionada ao processo de desenvolvimento utilizado. Processos bem definidos e maduros têm maior probabilidade de levar a produtos de qualidade. No entanto, a qualidade do software pode ser afetada por fatores externos, como pressão para lançamento antecipado ou mudanças constantes de requisitos. Portanto, a relação entre qualidade do processo e qualidade do produto é complexa e pode variar dependendo do contexto.

Além disso, a avaliação de atributos de qualidade, como confiabilidade e manutenibilidade, pode ser desafiadora e muitas vezes requer o uso do software por um longo período para obter resultados conclusivos. A qualidade do software é, portanto, um campo dinâmico e complexo, influenciado por fatores técnicos, contextuais e humanos. À medida que exploramos esses conceitos, percebemos que a busca pela qualidade é uma jornada contínua e adaptativa, seja em ambientes mais tradicionais ou ágeis.

Aula 04

Vamos discutir um pouco sobre padrões de software, enfocando principalmente suas três características principais. Começando pelo primeiro ponto, esses padrões capturam a sabedoria acumulada que possui valor para a organização. Eles se baseiam no conhecimento adquirido por meio de tentativa e erro, ao longo do tempo, sobre as melhores práticas e abordagens adequadas para a empresa. Incorporar esse conhecimento em padrões permite que a organização reutilize experiências passadas e evite cometer erros já conhecidos, contribuindo para a construção de produtos de software de maior qualidade.

Em segundo lugar, os padrões fornecem um referencial para definir o que se entende por qualidade em um contexto específico. Essa característica é particularmente relevante, uma vez que a qualidade do software é subjetiva e complexa. Usando padrões, é possível estabelecer uma base para avaliar se um nível aceitável de qualidade foi alcançado. O padrão serve como um ponto de referência para determinar se o software atende tanto aos requisitos explícitos quanto aos requisitos implícitos, como usabilidade, que são esperados de um software de alta qualidade.

A terceira característica dos padrões está relacionada à continuidade. Quando um trabalho é transferido de um indivíduo para outro, os padrões de software fornecem uma estrutura uniforme para essa transição. Isso significa que todos os engenheiros dentro da organização adotam práticas semelhantes, minimizando o esforço necessário para aprender um novo projeto. Os padrões garantem que as melhores práticas e lições aprendidas estejam disponíveis para todos, facilitando a disseminação do conhecimento e a consistência nos projetos.

É importante destacar que existem dois tipos de padrões na engenharia de software: padrões de produto e padrões de processo. Os padrões de produto estão relacionados às características específicas do software em desenvolvimento. Eles incluem padrões de documentação, formatação de código, estrutura de documentos de requisitos, entre outros. Por outro lado, os padrões de processo definem como as atividades de desenvolvimento devem ser conduzidas. Isso abrange desde a especificação do projeto até a validação e lançamento do software, englobando também práticas de controle de qualidade.

Um aspecto crucial a considerar é que diferentes tipos de software requerem diferentes abordagens de desenvolvimento. Portanto, os padrões devem ser adaptáveis para atender às necessidades de cada projeto. Não faz sentido prescrever uma única maneira de trabalhar se ela não for apropriada para o contexto do projeto ou da equipe envolvida. Os padrões de software devem ser aplicáveis ao produto sendo desenvolvido e ao processo utilizado, de modo a assegurar a qualidade e eficiência no desenvolvimento.

Em suma, os padrões de software desempenham um papel fundamental na busca pela qualidade do software. Eles encapsulam experiências passadas, fornecem um referencial para avaliar a qualidade e facilitam a continuidade nos projetos. A adaptação dos padrões ao contexto é essencial para garantir que as melhores práticas sejam aplicadas de maneira apropriada a cada situação.

Aula 05

Vou abordar a ISO 9001 para que você compreenda o contexto. A ISO 9001 faz parte de um conjunto internacional de padrões que são empregados no desenvolvimento de sistemas de gerenciamento de qualidade em diversos setores. Esse conjunto é conhecido como ISO 9000. Esses padrões são aplicáveis a uma variedade de organizações, abrangendo desde manufatura até serviços. A ISO 9001, em particular, é um dos padrões mais abrangentes e se aplica a organizações envolvidas no design, desenvolvimento e manutenção de produtos, inclusive software.

Inicialmente desenvolvida em 1987, a ISO 9001 não se trata de um padrão de desenvolvimento de software, mas sim de um arcabouço para a criação de padrões de software. Diferentemente da ISO 12207, que trata do ciclo de vida de software, a ISO 9001 tem foco em padrões de qualidade. Ela estabelece princípios gerais de qualidade, descreve processos de qualidade de maneira geral e define normas e procedimentos organizacionais que devem ser estabelecidos. Tudo isso é documentado em um manual de qualidade organizacional.

A ISO 9001 aborda processos principais, como os de entrega de produtos e de apoio. Os processos de entrega abrangem desde a aquisição de negócios até a produção, entrega, teste e suporte. Por outro lado, os processos de apoio envolvem questões como gerenciamento de configuração e gerenciamento de inventário, com aplicações tanto em engenharia de software quanto em outras áreas.

A aplicação da ISO 9001 tem sido amplamente reconhecida, com muitas organizações certificadas por ela. No entanto, é importante ressaltar que a certificação ISO 9001 não garante, por si só, que a qualidade do software produzido por uma empresa certificada seja superior à de uma não certificada. A ISO 9001 foca no estabelecimento de procedimentos de gerenciamento de qualidade e sua aplicação, sem necessariamente abordar as melhores práticas de desenvolvimento de software. Melhores práticas de desenvolvimento de software são abordadas pela ISO/IEC 12207, MPS-BR e outros padrões específicos.

Em resumo, a ISO 9001 é um padrão reconhecido e aplicado em muitas organizações para assegurar o controle de qualidade geral. Entretanto, ele não abrange especificamente as práticas de desenvolvimento de software. Para garantir a qualidade de software, é necessário combinar a ISO 9001 com padrões específicos de engenharia de software, como a ISO/IEC 12207 e o MPS-BR, que tratam diretamente de boas práticas de desenvolvimento. Portanto, embora a ISO 9001 contribua para o controle de qualidade organizacional, ela não é suficiente para garantir a qualidade de software em si.

Aula 06

Vamos falar sobre revisões e inspeções, que são atividades essenciais para garantir a qualidade de um projeto. Essas atividades estão inseridas na garantia de qualidade e são voltadas para verificar a qualidade dos entregáveis do projeto. Basicamente, envolvem a verificação do software, sua documentação e registros de processo, a fim de identificar erros, omissões e violações de padrões. Essa análise é uma parte crucial da garantia de qualidade.

No contexto das revisões da qualidade, estamos lidando com documentos produzidos durante o processo de desenvolvimento do software, como especificações, projetos, design, código, modelos de processo, planos de teste, procedimentos de configuração, padrões de processo e manuais de usuário. O objetivo das revisões é garantir consistência, completude e conformidade com os padrões estabelecidos, a fim de manter a qualidade.

As revisões de qualidade são realizadas antes da etapa de teste, e elas têm o propósito de melhorar a qualidade geral do software. É importante notar que essas revisões não são para avaliar o desempenho das pessoas envolvidas no desenvolvimento, mas sim para aprimorar os processos e evitar erros recorrentes. Através dessas revisões, os erros individuais são expostos para toda a equipe de desenvolvimento, promovendo uma cultura de aprendizado e melhoria contínua.

As avaliações de progresso, por outro lado, têm uma abordagem diferente. Elas comparam o progresso real do projeto com o planejado, levando em consideração fatores externos e possíveis mudanças de circunstâncias. Essas revisões podem levar a decisões de ajuste no desenvolvimento do software, inclusive interrompendo ou modificando drasticamente o projeto, se necessário.

É importante distinguir entre revisões de qualidade e avaliações de progresso. Enquanto as revisões de qualidade se concentram em garantir a conformidade e melhoria do processo, as avaliações de progresso lidam com o acompanhamento do desenvolvimento em relação ao planejado e a possíveis mudanças nas circunstâncias que possam afetar o projeto.

Vale ressaltar que as inspeções de programa também são uma parte crucial do processo. Elas envolvem revisões por pares, onde membros da equipe colaboram para encontrar bugs no programa em desenvolvimento. Essas inspeções complementam os testes, pois não requerem a execução do programa e podem ser uma maneira eficaz de identificar defeitos.

Estudos passados mostram que inspeções de programas informais são altamente eficazes na detecção de erros, alcançando taxas de detecção de defeitos superiores às dos testes unitários. Embora esses estudos tenham sido realizados antes da popularização das ferramentas de automação, hoje em dia, ferramentas como o Sonar Cube podem automatizar grande parte dessas atividades, tornando-as ainda mais eficientes e precisas.

Portanto, é evidente que tanto as revisões quanto as inspeções são atividades essenciais para garantir a qualidade do software ao longo de todo o ciclo de desenvolvimento. Essas práticas contribuem para a identificação precoce de erros e o aprimoramento contínuo dos processos, resultando em um produto final de maior qualidade e confiabilidade.

Aula 07

Vamos discutir um pouco sobre a importância das qualidades dentro dos métodos ágeis, como prometido. Nos métodos ágeis da engenharia de software, a ênfase recai na qualidade durante o processo de desenvolvimento, com uma forte concentração no código em si. Documentação e processos que não se relacionam diretamente ao desenvolvimento de código são reduzidos, e a comunicação informal entre os membros da equipe é valorizada em detrimento das comunicações baseadas em documentos formais.

Nosso estudo anterior dos fundamentos do Agile nos lembra dos valores que o sustentam. Isso não implica uma completa ausência de documentação, mas ressalta a entrega de produtos de alta qualidade e a redução de processos e ferramentas. A qualidade em métodos ágeis está intrinsecamente ligada à qualidade do código, com práticas como refatoração e Desenvolvimento Orientado a Testes (TDD) em foco. Essas práticas visam garantir que o código seja robusto e de alto padrão.

Refatoração é um processo constante de melhoria do código já implementado, incorporando ajustes incrementais conforme o software evolui. O TDD, por sua vez, é uma técnica que surgiu no movimento Extreme Programming (XP), onde o teste é considerado antes mesmo da escrita do código. A comunidade Agile desafia as abordagens mais burocráticas, como o padrão ISO 9001, enxergando o excesso de documentação como um fardo desnecessário. O gerenciamento da qualidade em métodos ágeis se baseia em boas práticas compartilhadas, em vez de processos formais e documentação.

É importante ressaltar que o Agile não prega a eliminação completa de processos e documentação, mas sim a utilização do que é essencial. As boas práticas para a qualidade de software em um ambiente ágil incluem verificar o código antes do check-in. Os programadores são responsáveis por revisões de código internas, assegurando que o código seja de qualidade antes de ser submetido ao sistema de construção. Além disso, a regra é "nunca quebrar a construção", o que significa que o código submetido não deve causar falhas no sistema como um todo.

No desenvolvimento ágil, o código não pertence a um indivíduo específico, mas sim ao time como um todo. Isso permite que qualquer membro do time faça melhorias no código implementado por outros, eliminando a noção de propriedade individual do código. Essa mentalidade colaborativa promove a inspeção contínua do código e evita a acumulação de débito técnico.

A prática de revisões de código, realizadas seja individualmente ou através de programação em pares (dois programadores trabalham juntos no mesmo código), é fundamental para a qualidade. A programação em pares permite uma inspeção constante do código, encontrando defeitos que podem passar despercebidos de outra forma. No entanto, há desafios nessa abordagem, como mal-entendidos mútuos e relutância em criticar o trabalho do colega.

Em relação a sistemas de longa vida útil ou projetos com equipes geograficamente distribuídas, pode ser necessário adaptar as práticas ágeis para incluir algum grau de documentação e processos formais de qualidade. Isso é especialmente verdadeiro quando se trata de sistemas desenvolvidos para clientes externos, que podem ter suas próprias expectativas e requisitos de qualidade. Por exemplo, empresas que desenvolvem software podem incorporar um papel de QA (Garantia de Qualidade) para garantir a aderência às práticas de qualidade.

Portanto, a abordagem ágil para a gestão da qualidade pode ser altamente eficaz em desenvolvimento interno, onde a empresa controla a especificação do software. No entanto, em cenários mais complexos, como sistemas distribuídos ou sistemas de longa vida útil, a adaptação das práticas ágeis para incluir alguma formalização pode ser necessária para atender às expectativas de qualidade de todos os envolvidos no processo. A ideia chave é que a abordagem ágil é flexível e deve ser ajustada para atender às necessidades específicas de cada projeto.

Aula 08

Vamos falar sobre medição de software. Se estamos lidando com processos, precisamos de controle, e para controlar, é necessário medir. A medição de software envolve quantificar atributos de um sistema de software, como complexidade e confiabilidade. Ao comparar os valores medidos com padrões da organização, podemos avaliar a qualidade do software e a eficácia dos processos, ferramentas e métodos.

Idealmente, o gerenciamento de qualidade utiliza medições de atributos que impactam a qualidade do software. Medir permite quantificar para avaliar. No entanto, na prática, não é tão simples. Existem vários tipos de medidas na medição de software, incluindo medidas diretas e indiretas. As medidas diretas são objetivas e incluem custo, esforço, linhas de código, velocidade de execução, memória, número de erros e a complexidade ciclomática.

Por outro lado, as medidas indiretas estão relacionadas à funcionalidade, qualidade, complexidade, eficiência, confiabilidade e manutenibilidade. No contexto da engenharia de software, as métricas são características mensuráveis de um sistema de software, da documentação ou do processo de desenvolvimento. Elas envolvem medidas e podem ser combinadas para obter resultados mais amplos.

Exemplos de métricas incluem o tamanho do produto em linhas de código, o índice FOG que mede a compreensão de um texto, o número de falhas relatadas em um software entregue, e o esforço necessário para desenvolver um componente do sistema. Métricas podem ser de controle, relacionadas a processos de software, ou de previsão, relacionadas ao próprio software. As métricas de controle auxiliam na gestão do processo, enquanto as de previsão ajudam a prever características do software.

A complexidade ciclomática é uma métrica antiga que mede a complexidade de um módulo pelo número de caminhos independentes que ele pode percorrer. Métricas podem influenciar decisões de gerenciamento e têm diferentes relações com atributos de qualidade. Métricas dinâmicas são medidas durante a execução do programa, como o número de defeitos relatados. Métricas estáticas são medidas em representações do sistema, como tamanho do código ou complexidade ciclomática.

A relação entre métricas e atributos de qualidade nem sempre é direta. Métricas podem ser usadas para atribuir valores aos atributos de qualidade, mas a relação pode ser complexa. A usabilidade, por exemplo, não pode ser diretamente associada ao tamanho do manual do usuário. As métricas dinâmicas avaliam eficiência e confiabilidade, enquanto as métricas estáticas avaliam complexidade, compreensibilidade e manutenibilidade.

Em resumo, a medição de software envolve quantificar atributos para avaliar a qualidade e eficácia dos processos, ferramentas e métodos. Métricas dinâmicas e estáticas auxiliam na avaliação da eficiência, confiabilidade e complexidade do sistema, bem como de seus componentes. Embora as métricas proporcionem indicadores, a relação com os atributos de qualidade pode ser complexa. Portanto, a medição é uma ferramenta importante, mas a interpretação exige consideração cuidadosa.