Universidade Federal de Pernambuco
Centro de Informática
Pós-Graduação em Ciência da Computação

Dissertação de Mestrado

Inspector: Um Processo de Avaliação de Progresso para Projetos de Software

por

Javé Barbosa de Meneses

Orientador
Prof. Hermano Perrelli de Moura

Recife, fevereiro de 2001

UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA

Javé Barbosa de Meneses

Inspector: Um Processo de Avaliação de Progresso para Projetos de Software

Dissertação submetida à avaliação como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

ORIENTADOR
Prof. Hermano Perrelli de Moura

Recife, fevereiro de 2001

Dedico este trabalho aos meus queridos pais que sempre me apoiaram e deram provas de carinho, respeito e dedicação.

Agradecimentos

Agradeço a Deus, fonte de toda a criação, que sempre me iluminou, trazendo inspiração e perseverança para realização deste trabalho. Ele, com sua imensa bondade, ofereceu-me a possibilidade de pesquisar e buscar novos conhecimentos como forma de ajudar a outras pessoas.

É impossível deixar de agradecer aos meus pais e irmãos que sempre estiveram ao meu lado nos momentos alegres e tristes por que passei nestes dois anos. Eles são as pessoas mais importantes para mim e, mesmo com toda a distância, sempre estiveram presentes, seja nas conversas por telefone ou no meu pensamento, me guiando e ajudando a ter força e dedicação para que conseguisse realizar meus objetivos.

Agradeço ao Prof. Hermano Moura que além de meu orientador, sempre mostrando o melhor caminho para realização do trabalho, tornou-se também um grande amigo. Agradeço também aos professores Augusto Sampaio e Alexandre Vasconcelos em nome de todos os professores do Centro de Informática que me apoiaram bastante e deram dicas importantes para o desenvolvimento do trabalho.

Agradeço aos companheiros de jornada Alessandro e Tiago, que nestes dois anos foram uma verdadeira família, compartilhando momentos de alegria e tristeza, saúde e doença. Espero que nossa amizade, iniciada há seis anos, permaneça para sempre. Agradeço também a todos os colegas e amigos do Centro de Informática, que participaram comigo de “farras” memoráveis e das famosas peladas no Inocoop.

Agradeço também ao Tai Mu Shi (China), Moisés e André Lucena, em nome de todos os funcionários da Emprel – Empresa Municipal de Informática de Recife, que me ajudaram bastante, fornecendo todas as informações necessárias para a aplicação prática do Inspector dentro da empresa.

Resumo

A competitividade e a existência de projetos de software cada vez maiores e mais complexos tornou a utilização de técnicas de gerenciamento uma prática fundamental para obtenção do sucesso de projetos de desenvolvimento de software. Nesse contexto, a mensuração de projetos de software representa um importante auxílio, quantificando propriedades que permitem controlar o processo de desenvolvimento e a qualidade do produto. Apesar do grande interesse nessa área nos últimos anos, ainda se observa a necessidade de aperfeiçoamento de algumas técnicas e ferramentas, de modo a permitir um monitoramento mais preciso do progresso técnico de um projeto.

Este trabalho busca contribuir para a obtenção de projetos de desenvolvimento de software mais controlados e gerenciados, definindo um processo de avaliação de progresso para projetos de software orientado a objetos. O Inspector, nome dado ao processo, define um conjunto de atividades que visam tornar o acompanhamento de projetos uma tarefa sistemática, onde os problemas no desenvolvimento são identificados antes que atinjam proporções maiores, e mecanismos são inseridos de modo a permitir que o gerente de projeto forneça uma avaliação precisa do *status* do projeto ao cliente.

O Inspector fornece ao gerente de projeto duas visões de progresso complementares que fornecem suporte para um bom gerenciamento do processo de desenvolvimento. São elas: visão de desempenho e visão de funcionalidade. A primeira visão verifica o desempenho das equipes de desenvolvimento a partir da análise das atividades planejadas para as mesmas, definindo três métricas que mostram a qualidade do planejamento e a produtividade da equipe de desenvolvimento. Já a visão de funcionalidade define uma métrica, dirigida a casos de uso, que verifica o progresso funcional do sistema. De posse das duas visões, o gerente de projeto é capaz de identificar equipes com dificuldades e casos de uso com problemas no desenvolvimento.

Foi realizado um estudo de caso que consistiu na aplicação do Inspector a um projeto real, de forma a validar as métricas de progresso, as atividades e os artefatos propostos. O processo como um todo busca garantir um bom uso de tais métricas, e o registro das experiências obtidas durante a realização do projeto. O estudo de caso introduziu melhorias no processo, revelou algumas limitações existentes no trabalho atual, permitindo também a identificação de trabalhos futuros que podem ser realizados para aperfeiçoar o processo.

Abstract

Global competition and the existence of software projects of increasing complexity and size turned the use of management techniques a fundamental practice for successful software development projects. In this context, measurement of software projects represents an important issue, allowing the quantification and control of the development process and the product's quality. In spite of the growing interest in that area in recent years, some techniques and tools are necessary to achieve some improvements, in a way to allow a more precise management of the project's technical progress.

This work contributes towards a more controlled and managed software development environment, defining a progress assessment process for object oriented software projects. Inspector, as this process is called, defines a set of activities which can be used to manage the project progress in a systematic way, where the problems in the development are identified before they produce larger consequences, and mechanisms are inserted which allow the project manager to report a precise assessment of the project status to the customer.

Inspector provides two complementary views of the project progress: the performance view and functionality view. The first view verifies the performance of the development teams from the analysis of the planned activities; three metrics are defined to show the quality of the planning and the productivity of the team. The functionality view, on the otherside, defines a metric driven by use cases that verifies the functional progress of the system. With these two views, the project manager is capable to identify teams with difficulties and problems in the development of the system.

The process was applied to a real project, in way to validate the proposed progress metrics, activities and artifacts. The defined process tries to guarantee a good use of such metrics and records the experiences obtained during the development of the project. This case study introduced improvements, and identified limitations in the current work and areas for further improvement for the Inspector process.

Conteúdo

Tabelas

Figuras

Capítulo 1

Introdução

No cenário econômico e tecnológico atual, as transformações no mercado são constantes. As organizações que buscam o sucesso empresarial devem estar preparadas para grandes desafios técnicos e o aproveitamento das oportunidades de negócios através de um processo contínuo de transformação e melhoria. A cada dia, torna-se mais importante a utilização de técnicas e conceitos de engenharia de software que surgem como aliados em um ambiente onde o amadorismo e a produção não controlada de software não se enquadram, cedendo lugar para o desenvolvimento sistemático e gerenciado, aumentando a qualidade dos produtos desenvolvidos e a produtividade dos membros da organização.

O gerenciamento de projetos de software não é uma tarefa trivial. Controlar grandes equipes de desenvolvimento de modo que o aproveitamento das mesmas seja satisfatório exige um acompanhamento próximo, além da utilização de técnicas e modelos de métricas que quantificam e qualificam o andamento do projeto [27]. Assim sendo, para se monitorar o desenvolvimento é importante acompanhar o progresso do projeto em andamento, comparando a situação atual do projeto com o estimado.

Nesse contexto, métricas de software tem se tornado um fator essencial de auxílio ao gerente de projeto na captura das informações relevantes para o gerenciamento da qualidade do produto e do processo de desenvolvimento, proporcionando um melhor entendimento das relações existentes entre as atividades e as entidades que as mesmas afetam: produto ou processo [47]. O estudo das métricas de software evoluiu bastante nos últimos anos, devido ao grande interesse de pesquisadores que buscam soluções e

padronizações para os diversos desafios e problemas existentes na área. Apesar dessa evolução, ainda é necessária a realização de muito trabalho para que a engenharia de software e, consequentemente, as métricas de software adquiram alicerce teórico suficiente para garantir a construção de um software que atenda a demanda dos usuários com a qualidade desejada e a produtividade esperada.

1.1 Motivação

Um desafio que persiste desde o começo do desenvolvimento comercial de sistemas de software, com os sistemas se tornando cada vez maiores e complexos, é a solução dos problemas de comunicação entre o cliente e a organização [30]. O que se percebe hoje, em grande parte dos casos, é que não existe um consenso entre as duas partes, tornando a relação entre ambos desgastada. O cliente, muitas vezes, não sabe expressar o que realmente deseja e, por sua vez, a organização geralmente não consegue mostrar ao cliente, de forma clara, os avanços realizados desde o início do projeto. Muitas vezes o cliente deseja saber a situação do projeto que ele está financiando e, nesse momento, é importante que a organização esteja preparada para dizer o que já foi feito em relação ao combinado, o quanto já se gastou até o momento, fornecendo estimativas precisas que indiquem quando o projeto será finalizado, e justificando, através de documentos concisos, os motivos de atrasos, caso eles existam.

A preocupação com a definição de atividades sistemáticas, que utilizam técnicas e métodos bem definidos como forma de se melhorar a produtividade e a qualidade dos produtos desenvolvidos, ganhou expressividade nos últimos anos e representa uma área em constante evolução [44]. Apesar disso, observou-se que alguns dos processos de desenvolvimento atuais, como por exemplo o RUP [7, 48] e o OPEN [24, 25], não cobrem, ou cobrem parcialmente, o acompanhamento de projetos de software. Nesse contexto, mesmo em organizações que possuem um processo bem definido, com atividades específicas para o desenvolvimento e padrões de documentação estabelecidos, é importante existir atividades voltadas para o gerenciamento de projetos, identificando claramente como recuperar informações sobre o progresso do projeto, avaliando e solucionando possíveis dificuldades que surjam.

O desenvolvimento de projetos de software, diferentemente da construção de um automóvel, não consiste em uma tarefa uniforme, pois envolve uma série de fatores externos que podem afetar o desenvolvimento, ocasionando atrasos em relação ao planejado. Além disso, a construção de um software envolve a participação efetiva de seres humanos (usuários ou membros da equipe de desenvolvimento) que, muitas vezes, apresentam comportamento e produtividade distintos, de acordo com o ambiente de trabalho e problemas pessoais. Desse modo, a observação e acompanhamento da produtividade da organização representa um importante fator, que poderá indicar ao gerente o sucesso de um determinado projeto e auxiliá-lo na estimativa de projetos futuros, a partir da utilização de dados de produtividade obtidos em projetos anteriores.

Além do acompanhamento da produtividade da organização, é importante também medir o quanto se progrediu em termos do processo de desenvolvimento utilizado, das atividades que devem ser realizadas e dos artefatos que devem ser construídos durante o desenvolvimento do sistema. Um dos desafios desse trabalho é a proposta de uma métrica única que indique o progresso funcional (ou técnico) do projeto, observando aspectos de diversas métricas de progresso já existentes. Além dessa métrica, o trabalho define um conjunto de três métricas que avaliam a produtividade e uma série de atividades que auxiliam a organização a fazer um bom uso das métricas definidas.

1.2 Definição do Problema

Uma importante pergunta que deve ser respondida nesse momento é: q*ual o problema que o trabalho procura resolver?*

A resposta a esta pergunta pode ser encontrada através da observação da situação atual do desenvolvimento de projetos em grandes organizações. O que se observa é a existência de atrasos relevantes no cronograma inicialmente previsto, e limitações no processo de gerenciamento que dificultam a identificação e tratamento de problemas, que em grande parte dos casos representam a causa do insucesso do projeto [27]. Apesar da preocupação com o gerenciamento de projetos ter aumentado substancialmente nos

últimos anos, alguns aspectos de gerenciamento de projetos não foram solidificados e necessitam de trabalhos para aperfeiçoamento e melhoria.

Nesse contexto problemático atual, percebe-se que alguns processos de desenvolvimento não cobrem com precisão o acompanhamento de projetos, e necessitam ser adaptados ou ampliados de forma a permitir um gerenciamento mais próximo do desenvolvimento, observando a(s) equipe(s) de desenvolvimento e o software que está sendo construído. Apesar de existir um grande conjunto de métricas para projetos de software orientado a objetos que observam a qualidade do produto e do processo de desenvolvimento [5, 13, 14, 23, 30, 35, 51], ainda não há um consenso da eficiência de tais métricas, e nem foi definido claramente um conjunto de atividades que permitam o acompanhamento de projetos através da utilização de tais métricas.

Um outro problema que existe atualmente em algumas organizações é a falta de um processo sistemático que mantenha as experiências obtidas pela organização em projetos anteriores. É muito importante reaproveitar conhecimento, de modo a fazer com que o gerente produza estimativas e planejamento cada vez mais precisos e próximos da realidade. Problemas de estimativa acarretam diversos outros problemas, como os apresentados por Queiroz [47]. Dentre os problemas citados, destacam-se a sobrecarga de trabalho sobre os membros da equipe de desenvolvimento, desgaste com o cliente devido a atrasos no tempo previamente acertado, redução do escopo do sistema em desenvolvimento devido a uma má estimativa de custo, e até mesmo a interrupção do desenvolvimento do sistema.

1.3 Objetivos Específicos

Este trabalho apresenta uma proposta para solucionar as dificuldades encontradas pelo gerente durante o acompanhamento e monitoramento do progresso de projetos de software orientado a objetos. Foi definido um conjunto de métricas de avaliação de progresso, divididas em métricas de desempenho e métricas de progresso funcional, que possibilitam a análise e tratamento estatístico da situação do projeto em um determinado momento. Além da definição desse conjunto de métricas de caráter específico, foi

definido também um processo de avaliação de progresso, que visa fornecer maturidade suficiente à organização, de modo a torná-la capaz de configurar e manipular as métricas definidas para um projeto específico, avaliar os resultados obtidos e solucionar os problemas encontrados. Desse modo, esse trabalho irá:

- Apresentar a importância do gerenciamento em um projeto de software, mostrando a necessidade de se monitorar o desenvolvimento de projetos através da utilização de métricas que permitam a realização de análises estatísticas e a observação de tendências no desenvolvimento.
- Garantir que sejam recuperados valores realísticos sobre os resultados atuais do projeto e o desempenho das equipes envolvidas, para que sejam comparados com os planos aprovados e documentados.
- Definir um conjunto reduzido de métricas para avaliação do progresso de um determinado projeto, classificando-as em métricas de desempenho ou de progresso funcional, de acordo com o indicador de progresso que ela quantifica.
- Definir um processo que garanta o bom uso das métricas definidas, adaptando a organização para que ela adquira maturidade suficiente para coleta, cálculo e avaliação destas métricas.
- Fornecer um conjunto de tarefas que auxiliem o gerente na tomada de decisão, para solucionar os problemas encontrados durante a avaliação.
- Realizar a aplicação prática do processo e das métricas em um estudo de caso, identificando as contribuições e limitações do trabalho realizado.

1.4 Metodologia

Essa seção descreve a metodologia empregada para o desenvolvimento deste trabalho. Basicamente, a realização do trabalho foi dividida nas seguintes etapas:

Etapa de Estudo Inicial

- Estudo geral de trabalhos na área de engenharia de software [7, 8, 33, 44], que mostrou uma visão geral de processos de desenvolvimento e das limitações existentes na engenharia de software.
- Estudo geral de trabalhos na área de gerenciamento de projetos [20, 27, 28], identificando áreas de possíveis trabalhos e limitações existentes.
- Definição do escopo do trabalho, que visa suprir os problemas de monitoramento de projetos de software identificados durante esse estudo inicial.

Etapa de Estudo Aprofundado

- Estudo aprofundado de trabalhos na área de métricas de software [5, 13, 14, 23, 30, 35, 51], gerenciamento de projetos [16, 20, 27, 44], e processos de desenvolvimento [7, 24, 25, 48], que serviram como a base acadêmica para a confecção deste trabalho.

Etapa de Definição das Métricas de Avaliação de Progresso

- Identificação das métricas de avaliação de progresso existentes [5, 23, 30, 35], observando as limitações e a cobertura das mesmas.
- Classificação do conjunto de métricas que deverão ser manipuladas durante a avaliação de progresso em métricas de desempenho, que avaliam a produtividade da equipe de desenvolvimento, e métricas de avaliação de progresso funcional, que observam a quantidade de funcionalidade prevista para o sistema já incorporada dentro do mesmo.
- Definição das métricas e técnicas que serão utilizadas para medir desempenho, indicando a sensibilidade, a cobertura e como tais métricas são calculadas.
- Definição de uma métrica única para avaliação do progresso funcional que identifica percentualmente o progresso técnico no desenvolvimento do projeto. A escolha de uma métrica única e configurável tem como objetivo tornar a avaliação de progresso mais prática, automática e de fácil avaliação.

Etapa de Definição do Inspector

- Definição dos papéis das pessoas envolvidas com o processo de avaliação de progresso, indicando as responsabilidades e tarefas de cada um.
- Identificação das principais atividades envolvidas com a inserção de uma cultura de utilização de métricas que torne a organização capaz de utilizar as métricas definidas na etapa anterior.
- Identificação e detalhamento das principais atividades envolvidas com a coleta, cálculo e avaliação do progresso técnico de um projeto de software. Definição dos passos que deverão ser realizados para execução da atividade, e dos artefatos que deverão ser produzidos e inspecionados durante a avaliação de progresso.
- Organização de todas as atividades e métodos em uma ordem lógica, e atribuição das atividades a seus respetivos responsáveis.

Etapa de Aplicação do Inspector

- Documentação do Inspector em uma *homepage*, permitindo acesso via *web* [37].
- Realização de um estudo de caso em uma empresa e projeto reais, visando experimentar e melhorar o processo definido. Foi acompanhado o *status* do projeto Nota Fiscal Virtual durante mais de dois meses.

Etapa de Documentação do Trabalho

- Análise e comparação dos resultados obtidos, identificação dos problemas enfrentados durante o estudo de caso e adaptação do Inspector visando superar os problemas encontrados.
- Redação da dissertação.

1.5 Estrutura da Dissertação

Descrevemos a seguir a estrutura da dissertação. O Capítulo 1 apresenta uma introdução geral sobre o trabalho, identificando a motivação para realização do mesmo, ou seja, qual problema ele pretende solucionar, e delimitando seu escopo e objetivos principais.

O Capítulo 2 apresenta o resultado de um estudo bibliográfico, abrangendo os conceitos e princípios relacionados com o gerenciamento de projeto, e mais especificamente com o acompanhamento de projetos de software, que corresponde ao foco principal do trabalho realizado. Nele foram discutidos aspectos de gerenciamento de projetos, o papel do gerente de projeto durante o desenvolvimento, as principais atividades necessárias para o gerenciamento e acompanhamento do progresso de um determinado projeto, além de ter sido realizada uma breve avaliação do gerenciamento em cada um dos cinco níveis de maturidade do CMM (*Capability Maturity Model*) [5, 42].

O Capítulo 3 descreve o estado da arte das métricas utilizadas para mensuração do software e seu processo de desenvolvimento. Apresenta o estudo bibliográfico realizado para definição das métricas de avaliação de progresso utilizadas no Inspector, identificando os principais conceitos e métricas conhecidos. Ele apresenta as propriedades desejáveis a uma métrica, para que ela seja realmente útil durante o desenvolvimento. A teoria da mensuração, definida em [3, 18, 54], que representa um formalismo para identificar se uma métrica é realmente apropriada, foi apresentada e servirá como base para fundamentar a validade da métrica de progresso funcional definida. Além disso, esse capítulo mostra também a classificação de métricas tradicionais e orientadas a objetos, citando exemplos de cada uma dessas classificações. Por fim, diversos *suites* de métricas, definidos por pesquisadores da área [1, 13, 14, 35, 55], são apresentados como fonte de informação.

O Capítulo 4 foi dedicado à definição da métrica de progresso funcional única que captura as funcionalidades inseridas dentro do projeto. Inicialmente ele apresenta as principais características da métrica definida e as facilidades que ela proporciona. Em seguida, é apresentada a maneira como os dados são calculados e como as diversas equações definidas podem ser configuradas para um projeto específico. As principais formas de representação dos valores obtidos pela métrica são apresentadas, indicando como os resultados podem ser avaliados visando a identificação de problemas no desenvolvimento. Ainda nesse capítulo, a métrica definida é validada segundo a teoria da mensuração, apresentada no Capítulo 3, de forma a garantir que a métrica apresenta

as propriedades matemáticas desejáveis a uma métrica confiável. Por fim, ele apresenta a cobertura da métrica definida em relação às métricas de avaliação de progresso existentes, e as limitações que esta métrica apresenta.

O Capítulo 5 consiste na definição do Inspector, indicando as principais propriedades do processo e apresentando como é feito o acompanhamento de projetos no mesmo. São apresentadas as duas visões de progresso do projeto (desempenho e funcionalidade) que o Inspector apresenta como resultado para o gerente, permitindo uma avaliação precisa do *status* do projeto. Além disso, apresenta como o processo é estruturado, os artefatos que devem ser produzidos, as atividades definidas, como essas atividades são realizadas, quem são os responsáveis pelas atividades, como as métricas definidas são coletadas, calculadas, e como avaliar os resultados encontrados. Por fim, são identificadas algumas limitações e considerações finais sobre o Inspector.

O Capítulo 6 apresenta o estudo de caso realizado sobre um projeto real em uma organização de desenvolvimento de software, visando experimentar e melhorar o processo definido. Foram executadas as atividades e construídos os artefatos providos no Inspector, analisando inicialmente a organização, inserindo uma cultura de utilização das métricas de desempenho e progresso funcional e, a partir disso, foram documentados os resultados obtidos na coleta, cálculo e avaliação de tais métricas, que visam identificar o progresso do projeto de software. Todos os artefatos, contendo os resultados obtidos e problemas identificados no desenvolvimento do projeto, foram repassados para a organização que pôde observar e validar como estava o progresso do projeto.

Finalmente, o Capítulo 7 apresenta a conclusão do trabalho, indicação de trabalhos futuros e considerações finais.

Capítulo 2

Gerenciamento de Projeto

Apesar do muito que já se evoluiu na área de desenvolvimento de sistemas, a realidade atual ainda é bastante limitada. A baixa aceitação do produto final (sistema) por parte do cliente é um fato marcante. Em sistemas comerciais somente 40% das versões finais dos sistemas são aceitas pelos usuários. Em sistemas críticos esse valor é um pouco melhor, com cerca de 75% dos sistemas sendo aceitos pelos usuários, mas ainda representa um resultado problemático [33]. Um dos motivos da baixa aceitação do produto pelos usuários finais é a falta de um processo de desenvolvimento gerenciado, que garanta a entrega do produto no tempo previsto e com a qualidade esperada.

A preocupação com o gerenciamento de projetos de software surgiu há poucas décadas, e tem se tornado cada dia mais relevante, representando um novo ramo da Engenharia de Software [20]. O aumento no interesse sobre as técnicas de gerenciamento de projeto é resultado de um crescimento na incerteza dentro do ambiente de negócio. Mudanças políticas, o avanço na tecnologia da informação, a globalização, entre outros aspectos, têm feito o universo dos negócios muito mais inseguro e de alto risco. As organizações não podem mais perder tempo, e deixar para amanhã o que se pode fazer hoje, sendo assim, nesse ambiente de mudanças, elas estão utilizando técnicas de gerenciamento como uma forma de garantir o sucesso de seus projetos.

As experiências de muitas organizações na implantação de técnicas de gerenciamento têm levado à seguinte conclusão: inserir conceitos e técnicas relativas ao gerenciamento de projetos na organização não é uma tarefa simples. Na realidade, as

mudanças culturais representam as maiores dificuldades pois não são facilmente aceitas por alguns membros da organização, que não acreditam nos benefícios que elas trarão para a organização [20].

Um outro problema que geralmente surge no desenvolvimento de sistemas de grande porte é o tempo. Conseguir produzir um software de qualidade, no tempo inicialmente previsto, não é uma tarefa simples. Sistemas de software que tiveram grande custo financeiro e de esforço para ser finalizado, tornam-se, muitas vezes, obsoletos, devido a excessiva demora na entrega da versão final do produto. Nesses casos, o gerenciamento é tarefa essencial, visando realizar um conjunto de atividades e tomadas de decisões, de forma a garantir que o projeto seja finalizado de maneira satisfatória no tempo mais curto possível.

As seções seguintes apresentam o papel do gerente durante o desenvolvimento de um projeto, os modelos de gerenciamento existentes, as principais preocupações de um processo de gerenciamento de projeto, enfatizando a importância de se monitorar o desenvolvimento, a relação entre o gerenciamento e os níveis de maturidade CMM, e, por fim, apresenta algumas considerações finais, argumentando o escopo do trabalho realizado, dentro da área de gerenciamento de projeto.

2.1 O Papel do Gerente

O papel do gerente de projeto consiste, sucintamente, em alcançar os objetivos do projeto, distribuindo o produto com alta qualidade, no tempo previsto e com o preço justo [27]. De acordo com a norma ISO 12207 de qualidade [28], que estabelece uma estrutura comum para os processos de ciclo de vida do software, o gerente é responsável pelo gerenciamento do produto, gerenciamento do projeto e gerenciamento das tarefas do(s) processo(s) aplicável(eis), tais como: aquisição, fornecimento, desenvolvimento, operação e manutenção de software, ou processos de apoio.

O gerenciamento de projetos de software, visando aumentar a competência da organização na identificação, avaliação, planejamento e execução de tais projetos, não é uma tarefa trivial. O gerente deve estar atento a um conjunto variado de aspectos, que

influenciam no desenvolvimento de um sistema, dentre os quais temos: aspectos cognitivos, relacionados aos fatores humanos envolvidos no projeto (equipe de desenvolvimento, *stakeholders*[1], etc.); aspectos econômicos, influenciados pelos interesses comerciais da organização; e aspectos técnicos, relacionados com as etapas e atividades relacionadas ao desenvolvimento do projeto (análise, projeto, implementação, teste, etc.) [6].

Tempo, qualidade e custo são as três principais variáveis que devem ser observadas no gerenciamento de um projeto [16]. Todas as propriedades restantes, desejáveis a um projeto, podem ser inseridas a partir destas três variáveis. Inevitavelmente, existe uma tensão entre estas variáveis: o tempo pode ser reduzido, mas somente aumentando os gastos (prejudicando o custo) ou diminuindo o potencial do sistema (prejudicando a qualidade), ou até mesmo prejudicando os dois. Da mesma maneira, a qualidade pode ser aumentada somente se o custo ou o tempo for prejudicado, ou ainda prejudicando os dois (Figura 2.1).

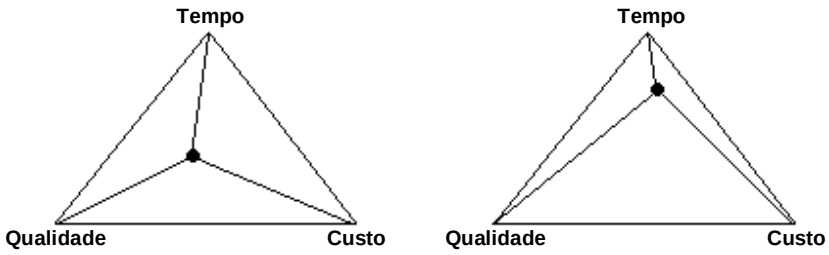


Figura 2.1. A relação entre tempo, qualidade e custo.

Ainda na Figura 2.1, temos que o primeiro diagrama ilustra a relação entre tempo, qualidade e custo. O que se observa é que, quanto mais um elemento é favorecido (representado através de uma linha mais longa), os outros dois elementos tornam-se mais prejudicados (representado através de uma linha mais curta). Desta forma, o segundo diagrama apresenta uma situação onde o tempo está sendo prejudicado (ou seja, será necessário grande tempo de desenvolvimento) visando garantir a qualidade do projeto. Assim, caso o tempo de desenvolvimento seja um fator de risco para o

[1] Qualquer indivíduo envolvido com o projeto que está sendo desenvolvido. [33]

desenvolvimento do sistema, o gerente de projeto deve tomar uma decisão para se reduzir o mesmo, resultando em um aumento na pressão sobre o custo ou qualidade.

Em ambientes onde as mudanças são frequentes, é muito comum haver uma reavaliação da relação tempo/custo/qualidade durante o desenvolvimento. Essa preocupação ocorre diversas vezes, à medida em que o projeto vai progredindo. O gerente deve garantir um balanceamento adequado destas três variáveis, de forma a alcançar um equilíbrio que garanta o sucesso do projeto.

2.2 Modelos de Gerenciamento de Projeto

Um modelo de gerenciamento representa o modo como a organização distribui as responsabilidades de gerenciamento entre os gerentes e os membros das equipes de desenvolvimento do projeto. Basicamente, existem três modelos de distribuição utilizados para o gerenciamento de projetos, dois dos quais o gerente de projeto desempenha papel semelhante [16]. O primeiro coloca o gerente no centro do processo de gerenciamento de um determinado projeto, coordenando as atividades entre times ou indivíduos (Figura 2.2).

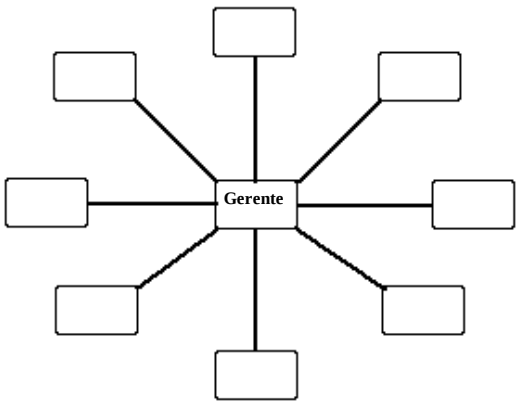


Figura 2.2. Gerenciamento de projeto centralizado

O problema deste modelo é a sobrecarga de responsabilidade sobre o gerente do projeto. Devido à falta de comunicação entre as diversas equipes dentro do projeto, todas as decisões importantes e trocas de informações necessárias entre as equipes,

devem passar pelo gerente de projeto, que pode facilmente perder o controle do processo de gerenciamento de projeto.

A Figura 2.3, representa um segundo modelo de gerenciamento alternativo, que reflete a hierarquia organizacional tradicional. Este modelo reconhece mais claramente a autoridade do gerente de projeto, porém ele implica que as decisões sejam tomadas na "raiz da árvore", e então comunicadas para os demais membros do projeto. Esta não é uma característica satisfatória, desde que ela ignora o fato que os especialistas em áreas específicas estão mais próximos das "folhas da árvore" de hierarquia, e que decisões são tomadas, preferencialmente, através da negociação destes especialistas com o gerente.

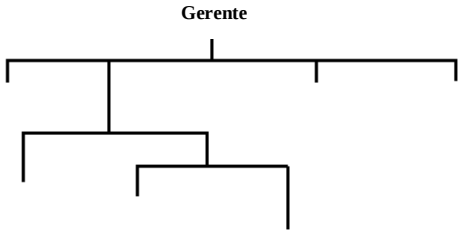


Figura 2.3. Modelo hierárquico do gerenciamento de projeto

A Figura 2.4 apresenta um modelo que reflete as melhores características dos dois modelos anteriores. Nesse caso, o gerente de projeto está claramente de posse de sua autoridade e, além disso, ocorrem comunicações através de relatórios informais (ou até formais) entre outras equipes, sem referências ao gerente de projeto. Este modelo deve ser utilizado de maneira cautelosa, já que o gerente do projeto deve ter conhecimento de todos os aspectos importantes do projeto que ele está envolvido. Encontrar o nível de comunicação entre as equipes de um determinado projeto leva tempo e deve ser feito cuidadosamente.

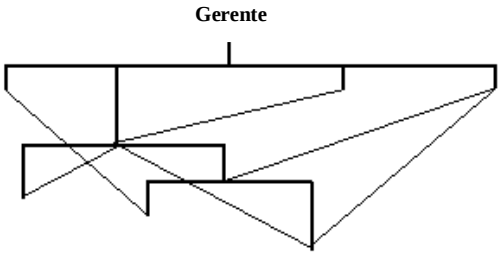


Figura 2.4. Modelo hierárquico com compartilhamento de informação entre equipes

2.3 Processo de Gerenciamento

Um processo de gerenciamento contém um conjunto de atividades e tarefas genéricas, bem definidas, que devem ser realizadas para garantir que um projeto seja bem gerenciado. As subseções a seguir, apresentam algumas das principais atividades envolvidas no gerenciamento de projeto, visando garantir o sucesso na execução de um projeto específico de acordo com diversos autores [20, 27, 44] e com a norma ISO 12207 [28].

2.3.1 Definição do Escopo

Antes que um projeto possa ser planejado, os objetivos e o escopo do projeto devem ser estabelecidos, soluções alternativas devem ser consideradas e, as restrições administrativas e técnicas identificadas [44]. Sem essas informações, é impossível definir: estimativas de custo razoáveis e precisas, uma divisão realística das tarefas de projeto ou uma programação de projeto administrável que ofereça indícios significativos de progresso.

O desenvolvedor de software e o cliente devem se reunir para definir os objetivos e o escopo do projeto. Os objetivos identificam as metas globais de projeto, sem levar em consideração como essas metas serão atingidas. O escopo identifica as funções primárias que o software deve realizar e, o que é mais importante, tenta delimitar essas funções de uma forma quantitativa [27].

Logo que os objetivos e o escopo do projeto forem compreendidos, soluções alternativas serão consideradas. Mesmo que poucos detalhes sejam considerados nesse momento, as alternativas possibilitam que gerentes e profissionais selecionem a abordagem mais adequada, dadas as restrições impostas sobre prazos de entrega, orçamento, disponibilidade de pessoal, interfaces técnicas e uma infinidade de outros fatores.

2.3.2 Planejamento

Uma das principais características para uma organização obter sucesso no gerenciamento de projetos, é a existência de planejamento [44]. Para resolver os conflitos existentes e estabelecer um *framework* para realização de tarefas, deve ser realizado um planejamento de operação anual. Nele são especificadas as tarefas a serem realizadas e atribuídas as responsabilidades e recursos para a realização das mesmas. Este plano lida com objetivos técnicos e de negócios em termos organizacionais. Desta forma, gastos, necessidades de novos capitais e questões de distribuição do produto são estabelecidos, periodicamente, por cada entidade organizacional. Além disso, objetivos de produtividade e eficácia podem ser estabelecidos, junto com estratégias para se alcançar os mesmos [27].

O planejamento de projeto, por sua vez, focaliza as atividades e objetivos de cada projeto, individualmente. Ele se preocupa principalmente com a funcionalidade, o custo, o cronograma e a qualidade, junto com os recursos e marcos de referência relacionados [26]. Enquanto cada projeto tem seu próprio gerenciamento e alguns recursos dedicados, todos os projetos contém recursos comuns para algumas atividades específicas. Eles também mantém relatórios para o Gerente Senior, em uma faixa periódica. Tipicamente, tais planos são monitorados por um grupo responsável, que informa problemas de gerenciamento e riscos encontrados para que se possa atingir as metas do projeto e da organização [27].

A distinção entre plano de período e de projeto geralmente é motivo de confusão. Na visão do pessoal envolvido em um determinado projeto, o trabalho dele é que é fundamental para os negócios da organização, sendo que ele não acredita ser necessário a informação periódica (plano de período). A organização entretanto, geralmente é medida e gerenciada sobre uma base periódica de tempo, e os dados do projeto devem ser traduzidos em termos de período, para serem incluídos em objetivos e planos anuais [26].

2.3.3 Analisando Riscos

A implementação de uma solução é baseada em planos (de projeto e de período) que contém estimativas, aproximações, incertezas e, consequentemente, envolve riscos. A identificação dos riscos existentes para realização do projeto, bem como o estudo de maneiras para evitá-lo e amenizá-lo são tarefas essenciais para o gerenciamento de qualquer projeto de software [20]. Dessa forma, a idéia de risco é bastante importante e deve ser considerada para evitar o desastre de um projeto.

'Um dos mais rigorosos teoremas econômicos prova que o significado existente de produção indica que uma melhor performance econômica só é alcançada através de um aumento no grau de incerteza, ou seja, elevando-se os riscos. Enquanto é uma tarefa inútil tentar eliminar todos os riscos, e questionável minimizá-los, é essencial que os riscos selecionados sejam os riscos certos...

Nós devemos estar dispostos a escolher racionalmente os riscos e as ações associadas, ao invés de identificar as incertezas baseados na intuição, não importa o quanto os riscos serão quantificados'

Peter Drucker (1975), *Management*, Heinemann

A análise dos riscos é composta por quatro atividades distintas: identificação, projeção, avaliação e administração dos riscos [44]. Cada atividade dessa é brevemente descrita a seguir:

- **Identificação dos riscos:** envolve a relação e classificação dos riscos específicos do projeto dentro das categorias de risco de projeto, técnico e de negócio. Os riscos de projeto identificam problemas orçamentários, de cronograma, de pessoal, de recursos, de cliente e de requisitos, e avaliam o impacto dos mesmos sobre o projeto de software em desenvolvimento. Os riscos técnicos ocorrem porque um problema é mais difícil de ser resolvido do que se imaginava. Os riscos de negócio estão mais voltados para o mercado e tendências fora da organização, podendo destruir os resultados dos melhores projetos de software.
- **Estimativas dos riscos:** classifica cada risco de acordo com a probabilidade de que ele seja real, e com as consequências dos problemas associados ao risco,

caso ele ocorra. Os riscos são ponderados em função do possível impacto percebido sobre o projeto, e depois colocados em ordem de prioridade.

- **Avaliação dos riscos:** durante esse processo de avaliação, é examinada, mais detalhadamente, a precisão das estimativas que foram feitas durante a projeção dos riscos, a partir de então, começa-se a observar maneiras de controlar e/ou evitar riscos que têm a probabilidade de ocorrer.
- **Monitoração dos riscos:** o trio (descrição, probabilidade e impacto dos riscos), associado a cada risco, é usado como uma base, a partir da qual os passos de gerenciamento dos riscos (ou aversão a riscos) são desenvolvidos.

2.3.4 Estimando Recursos

A estimativa dos recursos exigidos é feita para se levar em consideração o esforço de desenvolvimento do software. A Figura 2.5 ilustra os recursos de desenvolvimento como uma pirâmide. Cada recurso é especificado segundo quatro características: descrição do recurso, uma declaração da disponibilidade, tempo cronológico em que o recurso será exigido (data início de utilização recurso) e por quanto tempo o recurso será aplicado (intervalo de tempo no qual o recurso será utilizado no projeto) [44].

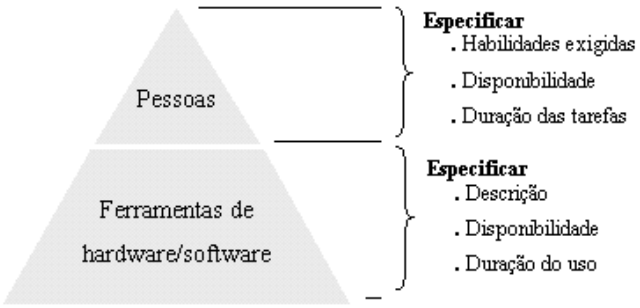


Figura 2.5. Recursos necessários para o desenvolvimento

O gerente deve avaliar o escopo do projeto e identificar as habilidades exigidas para concluir o desenvolvimento. A partir da necessidade de pessoal estimada, o gerente de projeto deve fazer um mapeamento do pessoal disponível para o projeto com o número de pessoas capacitadas que ele avaliou ser necessário [27]. Além disso, ele deve considerar três categorias de hardware durante o planejamento do projeto: hardware de desenvolvimento (usado para construir o software), hardware de produção ou operação

(no qual o software será executado) e elementos de hardware do novo sistema (recursos extras necessários para desenvolver e utilizar o novo sistema. Por exemplo: impressora, leitora óptica). Outra tarefa do gerente, nesse contexto, é avaliar os recursos de software necessários para desenvolver o projeto, ou seja, ele tem que identificar os softwares e ferramenta de suporte (CASE) necessários para que o sistema seja construído com sucesso.

2.3.5 Monitorando o Desenvolvimento

A atividade de monitoração e controle inicia-se logo que o planejamento do desenvolvimento for concluído. Cada tarefa, previamente planejada e identificada no plano de desenvolvimento é, então, rastreada pelo gerente de projeto [44]. O gerente pode usar uma ferramenta de planejamento e controle de projetos, automatizada, para determinar o impacto do não cumprimento dos prazos sobre os marcos de referência intermediários do projeto e a data de entrega global. Recursos podem ser redirecionados, tarefas reordenadas e compromissos de entrega modificados para acomodar o problema que foi descoberto, resultando em um desenvolvimento de software mais controlado.

O desenvolvimento de um projeto de software é monitorado através da identificação e coleta de métricas que auxiliam a avaliar se aspectos de cronograma, custo e qualidade têm sido desenvolvidos de acordo com o planejado [30]. Desse modo, é necessária a existência de uma disciplina de utilização de métricas que permita identificar, coletar e avaliar informações relevantes, que indiquem o andamento do projeto. De posse de tais métricas e das estimativas inicialmente realizadas, o gerente de projeto pode avaliar a situação atual, e verificar tendências que indiquem a existência de problemas futuros.

O Inspector foi definido justamente para suprir essa fatia do gerenciamento, identificando um conjunto de pessoas envolvidas, atividades que devem ser realizadas, documentos que devem ser produzidos, como forma de se acompanhar o desenvolvimento de sistemas de software. Desse modo, o trabalho realizado define um processo que visa a avaliação do progresso de um projeto orientado a objetos,

identificando o que se foi feito, o que se falta fazer, e verificando se problemas aconteceram, causando atrasos para o projeto.

2.4 O Acompanhamento de Projetos em Processos de Desenvolvimento

Uma das principais motivações para realização deste trabalho, ou seja, para definição de um processo para avaliação de progresso de projetos de software orientado a objetos, foi a observação que, a maioria dos processos de desenvolvimento atuais, não apresenta, de forma concreta e objetiva, como o gerente deve acompanhar o desenvolvimento de um projeto. Essa seção apresenta, brevemente, dois processos de desenvolvimento (RUP, OPEN) observados, e um processo de apoio ao desenvolvimento (PSP), mostrando até que ponto, o acompanhamento de projeto é suportado pelos mesmos.

2.4.1 *Rational Unified Process* – RUP

O Processo Unificado da Rational [7], conhecido como RUP, representa um processo genérico de desenvolvimento de software orientado a objetos, que pode ser especializado de acordo com o porte, área de aplicação e tamanho do projeto, além da maturidade da organização. As principais características desse processo são:

- **Dirigido a caso de uso[2]:** o processo segue um fluxo de ações para realização dos casos de uso. Assim, os casos de uso são especificados, projetados e utlizados como fonte para definição dos casos de teste. Eles dirigem todo o processo de desenvolvimento.
- **Centrado na arquitetura:** O conceito de arquitetura de software engloba os aspectos estáticos e dinâmicos mais significantes do sistema, tais como a plataforma

[2] Sequência de ações que incorpora uma determinada funcionalidade ao sistema, fornecendo a quem iniciou sua execução, um resultado que pode ser quantificado.

de software, componentes que podem ser reutilizados, sistemas legados e requisitos não funcionais. No RUP, o sistema é desenvolvido a partir da arquitetura definida.

- **Iterativo e incremental:** O desenvolvimento de um software, no RUP, é dividido em pequenos ciclos ou mini-projetos. Cada ciclo é uma iteração, que implica na realização de um conjunto de atividades definidas, e resulta em um incremento nas funcionalidades do sistema.

O gerenciamento de projetos no RUP, é realizado através da execução de um conjunto de atividades de apoio, definidas e agrupadas dentro de um fluxo, denominado Fluxo de Gerenciamento de Projeto. Segundo Booch [7], as atividades definidas, visam fornecer diretrizes práticas para organização planejar, prover de pessoal, executar e monitorar projetos.

Observando o RUP, percebe-se que o mesmo contém um conjunto bem definido de atividades para o planejamento do projeto, alocação de recursos e tratamento de riscos. Apesar disso, temos que o acompanhamento de projetos e a utilização sistemática de técnicas e métricas para monitorar o desenvolvimento deixam a desejar. O RUP não define um conjunto de tarefas que devem ser realizadas para um acompanhamento preciso do projeto, ele somente indica a importância em se realizar o mesmo. Ele apresenta um conjunto de diretrizes, que citam algumas métricas que podem ser usadas durante o desenvolvimento, para verificar o tamanho e a qualidade do produto que está sendo gerado, mas ele não apresenta um conjunto bem definido de métricas que possam ser utilizadas eficientemente para monitorar o progresso, nem visualizar o desempenho das equipes durante o desenvolvimento de um projeto.

2.4.2 Personal Software Process – PSP

O PSP [26] representa uma tecnologia desenvolvida pelo *Software Engineering Institute* (SEI), que define uma disciplina de trabalho para o engenheiro de software, representando uma estratégia para o auto-desenvolvimento profissional e melhoria de produtividade. Ele mostra como aplicar métodos de engenharia de software avançados, para os membros das equipes realizarem suas tarefas diárias. Fornece métodos detalhados de como se estimar e planejar tarefas, além de indicar como rastrear o

desempenho do indivíduo contra estes planos, e explicar como processos definidos podem guiar seu trabalho.

O PSP cobre bem o acompanhamento de projetos, definindo métricas e técnicas para monitorar o desempenho do indivíduo, o progresso das atividades que ele realiza e a qualidade do produto que está sendo desenvolvido. Algumas das técnicas que ele indica para monitorar o desenvolvimento são: rastreamento do tempo gasto no desenvolvimento, gerenciamento de cronogramas através de uma comparação do planejado com o realizado, métricas para estimar e verificar o tamanho do sistema, além de um conjunto extenso de técnicas para encontrar e remover defeitos, visando a qualidade do produto final.

Apesar de conter um conjunto bastante vasto e detalhado de métodos e métricas para acompanhamento de projetos, o PSP focaliza o desenvolvimento de um ponto de vista do indivíduo, membro da equipe de desenvolvimento. Dessa forma, ele não apresenta em detalhes, como agrupar os resultados individuais de cada membro, para acompanhar o progresso e a qualidade do projeto como um todo. Além disso, para o acompanhamento do tamanho do sistema, ele indica a métrica número de linhas de código (LOC) [26], que consiste em uma métrica relativamente problemática, onde nem sempre sistemas com um grande número de linhas de código, representam sistemas de qualidade [23].

2.4.3 Object-oriented Process, Environment and Notation – OPEN

O OPEN é um processo de terceira geração que focaliza o desenvolvimento orientado a objetos, e é construído ao redor de um meta-modelo bem definido, que consiste na arquitetura do processo de engenharia de software representado pelo OPEN [25]. Ele possui um ciclo de vida dirigido a contrato, onde as atividades definidas visam cumprir os contratos acertados com o cliente. Cada atividade é iniciada somente quando um conjunto de pré-condições são aceitas, e é composta de tarefas que devem ser realizadas para atingir o objetivo da atividade [23].

O OPEN apresenta um conjunto de tarefas que se preocupam em planejar e gerenciar o desenvolvimento de um projeto. A tarefa *Desenvolver e Implementar Plano de Alocação de Recursos* apresenta um conjunto de técnicas que podem ser usadas para o gerenciamento, indicando em quais situações elas podem ser empregadas. Algumas das técnicas para acompanhamento de projeto que ela cita são a utilização de: cronogramas para acompanhamento das atividades, gráficos PERT para monitorar a dependência entre as atividades, estimativa de custo, rastreamento, análise do trabalho realizado, entre outras [25]. Além disso, o OPEN identifica a necessidade de se configurar um conjunto de métricas que indiquem tanto a qualidade quanto o progresso do produto em desenvolvimento, citando o paradigma Goal-Question-Metric (GQM) [52] como forma de identificar tais métricas, e até mesmo o uso do PSP [26], apresentado na Seção 2.4.2, como processo de gerenciamento de pessoal que pode ser utilizado.

O OPEN apresenta uma sub-tarefa, que visa somente empregar métricas para o acompanhamento do processo de desenvolvimento e do produto que está sendo desenvolvido. Apesar disso, essa sub-tarefa não cobre completamente a complexidade envolvida em um processo de acompanhamento de projetos. Ela apresenta superficialmente um conjunto de técnicas e métodos que poderiam ser usados como forma de auxílio ao gerenciamento de projeto. Existe uma distância substancial entre o que é mostrado e a sua real aplicação, pois não são apresentados passos bem definidos, que demonstram como as métricas e técnicas podem ser empregadas.

2.5 O Gerenciamento e os Níveis de Maturidade

O CMM (*Capabilty Maturity Model*) [42] é um modelo descritivo e normativo que descreve os atributos essenciais que são esperados para caracterizar o nível de maturidade de uma organização particular. Este modelo fornece um *framework* para realização de um processo contínuo de melhoria dentro da organização, e se divide em cinco níveis de maturidade que definem uma escala ordinal para medir a maturidade e a capacidade do processo de desenvolvimento de uma organização.

Os cinco níveis de maturidade, mostrados na Figura 2.6, priorizam a realização de ações de melhoria para aumentar a maturidade do processo de desenvolvimento de software. As setas apresentadas na figura indicam o tipo de capacidade que está sendo inserido na organização em cada nível de maturidade.

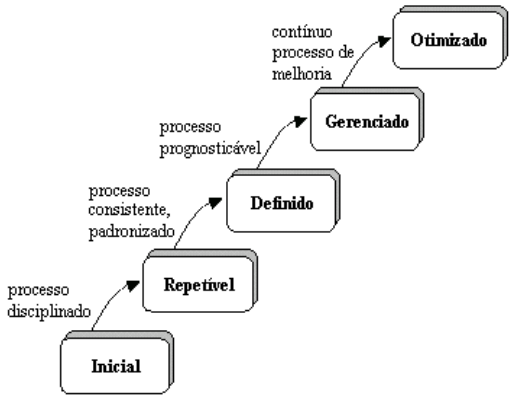


Figura 2.6. Os cinco níveis do processo de maturidade de software

A seguir, são apresentadas, resumidamente, as principais características de cada nível de maturidade, fazendo um prognóstico do gerenciamento de projeto em cada um destes níveis.

Nível 1 – Inicial

No nível inicial, o processo de desenvolvimento é caracterizado como *ad hoc*, podendo facilmente chegar ao caos [42]. Poucos processos são definidos e o sucesso depende de esforços individuais. Práticas de gerenciamento não são utilizadas, enfraquecendo os benefícios do uso de uma boa engenharia de software, devido a um planejamento não efetivo e a utilização apenas de técnicas dirigidas a reação, ou seja, visando corrigir erros somente depois que eles acontecem.

O sucesso do projeto depende, inteiramente, da experiência do gerente e de uma equipe de software eficaz, pois não existem normas e técnicas que devem ser seguidas para garantir um bom desenvolvimento [5]. Mesmo quando a organização possui gerentes capazes e experientes, o sucesso do projeto não é garantido, pois quando ele se desliga do projeto, sua influência sobre o mesmo também vai embora. Além disso, a

existência de um processo de engenharia de software não consegue suportar a falta de boas práticas de gerenciamento.

Organizações com capacidade de nível inicial têm seu processo de software constantemente alterado, à medida que o trabalho progride. Cronogramas, objetivos, funcionalidades e a qualidade do produto não são estimados devido à instabilidade do processo. Outra característica, é que o desempenho depende da capacidade de cada indivíduo e varia de acordo com sua habilidade, conhecimento e motivação.

Nível 2 – Repetível

No nível repetível, são estabelecidas políticas para gerenciar projetos de software e procedimentos para implementar estas políticas. O planejamento e gerenciamento de novos projetos baseia-se na experiência obtida com projetos similares [42]. Um objetivo atingido neste nível é permitir que organizações repitam com sucesso, práticas desenvolvidas em projetos anteriores, apesar que, práticas específicas para o projeto podem ser definidas.

Neste nível, os projetos têm controles básicos de gerenciamento estabelecidos, baseados nos resultados observados em projetos anteriores e nos requisitos do projeto atual. Existe a preocupação com o custo, os cronogramas e as funcionalidades do projeto. O gerente de projeto procura identificar os problemas que estão surgindo, utilizando técnicas para solucioná-los [5]. Armazenam os resultados obtidos no projeto, para utilização dos dados como forma de comparação em futuros projetos, através da definição de padrões para desenvolvimento e documentação do software.

A capacidade das organizações deste nível pode ser resumida como disciplinada, pois o desenvolvimento é suportado por um controle efetivo, garantido através de um planejamento e rastreamento estável do projeto de software, seguindo planos realísticos baseados no desempenho em projetos anteriores.

Nível 3 – Definido

No nível definido, o processo padrão para desenvolver e manter o software é documentado, incluindo os processos de engenharia e gerenciamento integrados em um

ambiente coerente [42]. Processos estabelecidos neste nível são usados para ajudar gerentes de software e pessoal técnico a realizar suas atividades mais efetivamente.

A organização utiliza práticas de software efetivas para padronizar seus processos de desenvolvimento. Ela possui um conjunto de processos de engenharia de software e de gerenciamento bem definidos, contendo: entradas, padrões e procedimentos para realizar o trabalho, mecanismos de verificação, saídas e critérios de verificação. No nível definido, as técnicas de gerenciamento de projeto resultam em uma boa compreensão do progresso técnico sobre todos os projetos.

Organizações com capacidade de nível definido podem ser resumidas como sendo padrão e consistente, porque tanto atividades de engenharia de software quanto de gerenciamento de projeto são estáveis e de fácil utilização. O custo, cronograma, código e funcionalidade do software que está sendo desenvolvido, estão sob controle, além disso, a qualidade do produto final é rastreada para garantir a satisfação do cliente.

Nível 4 – Gerenciado

No nível gerenciado, a organização configura metas de qualidade quantitativas para os produtos e o processo de software. A produtividade e a qualidade de um projeto são medidas, através de atividades definidas no processo de software, como parte de um programa organizacional de medição [42]. Um banco de dados é usado para coletar e analisar os dados disponíveis do projeto, em um processo de software definido. No nível gerenciado, tais processos são incrementados com métricas bem definidas e consistentes. Estas métricas estabelecem uma base quantitativa para avaliar os produtos e processos de software.

Projetos atingem o controle sobre seus produtos e processos, através do acompanhamento do desempenho do processo, que é classificado em limites quantitativos aceitáveis. Variações significativas no desempenho se diferenciam de variações aleatórias, especialmente dentro de aspectos de um produto bem estabelecido [5]. Os riscos envolvidos na inserção de alguma nova ferramenta ou técnica, dentro da organização, são identificados e gerenciados cuidadosamente.

Organizações com maturidade de nível gerenciado podem ser resumidas como prognosticáveis, pois o processo é fortemente medido, observando tendências na qualidade do processo, através de valores quantitativos, dentro dos limites definidos. Quando os valores excedem o limite, ações são tomadas para corrigir a situação. Como resultados desse processo sistemático de acompanhamento de projeto, temos que os produtos finais são geralmente de alta qualidade.

Nível 5 – Otimizado

No nível otimizado, toda a organização está concentrada em um processo contínuo de melhoria. Ela tem meios de identificar as fraquezas e qualidades do processo, com o objetivo de prevenir a ocorrência de defeitos. Dados sobre a efetividade do processo de software são utilizados para realizar análises de custo benefício das novas tecnologias e mudanças propostas para a organização [42]. Inovações que exploram as melhores práticas de engenharia de software são identificadas e inseridas na organização.

Organização com maturidade de nível otimizado apresenta um processo de gerenciamento que visa a melhoria contínua da maturidade da organização, acarretando em projetos com desempenho cada vez melhor. As melhorias ocorrem através de avanços incrementais no processo e de inovações usando novas tecnologias e métodos.

2.6 Considerações Finais

O capítulo mostrou os principais conceitos relacionados com o gerenciamento de projetos de software, considerando os aspectos desejáveis a uma organização para desenvolvimento de software de qualidade. Ficou clara a complexidade do processo de gerenciamento de projetos de software. É importante saber distribuir, entre os diversos gerentes e membros da equipe de desenvolvimento, a responsabilidade de se gerenciar o processo de desenvolvimento e o produto, de forma a não sobrecarregar o trabalho em cima do gerente, e permitir que o mesmo perca o controle das atividades do projeto.

Foram apresentadas, brevemente, a situação do acompanhamento de projetos em dois processos de desenvolvimento (RUP e OPEN), e como esse acompanhamento é

suportado pelo PSP, que é um processo de apoio ao desenvolvimento. Observou-se que nos processos de desenvolvimento, apesar de existir a preocupação, e se ter levantado a importância de se realizar um gerenciamento do progresso do projeto, ainda não foram definidas atividades claras que apresentem, em detalhes, uma forma sistemática de acompanhar e medir o desenvolvimento de projetos de software [7, 24, 25, 48]. O PSP, ao contrário, apresenta um grande conjunto de métricas que visam avaliar a maturidade no desenvolvimento, do ponto de vista do indivíduo, mas apresenta também métricas que auxiliam no gerenciamento do projeto como um todo [26]. A única dificuldade observada foi que as métricas definidas por ele muitas vezes não expressam de forma precisa o progresso no desenvolvimento de um determinado projeto.

O capítulo ainda apresentou os cinco níveis de maturidade definidos pelo modelo CMM [42], que visam implantar um processo de desenvolvimento gerenciado e de qualidade. Os principais aspectos do gerenciamento de cada nível, foram apresentados como forma de indicar as preocupações que a organização deve ter para conseguir alcançar um determinado nível de maturidade.

É perceptível que, por ser relativamente nova, a área de gerenciamento de projetos de software ainda é carente em diversos aspectos. As organizações dificilmente conseguem implantar com sucesso todas as técnicas e modelos apresentados neste capítulo. Geralmente, não existe um processo de rastreamento que permita a comparação de projetos anteriores executados com sucesso. Os padrões existentes são incompletos e, muitas vezes, não fornecem documentação suficiente para implantação adequada dos processos que elas apresentam.

A utilização de métricas durante o processo de gerenciamento, ainda não representa uma realidade para a maioria das organizações. A falta de uma cultura de acompanhamento precisa, impossibilita que o gerente identifique problemas futuros, fazendo com que a organização utilize uma política de reação a eventos, onde as ações tomadas pelo gerente visam minimizar ou eliminar problemas que já ocorreram dentro do sistema. Esse tipo de política não é adequado e, muitas vezes, acarreta o fracasso do projeto.

O presente trabalho realizado trata as dificuldades existentes no monitoramento do progresso de projetos, definindo um processo que permite avaliar a situação atual do desenvolvimento em termos do progresso atingido pelo mesmo. O progresso do sistema é avaliado a partir do desempenho das equipes envolvidas no projeto e do progresso funcional do sistema, que identifica a incorporação das funcionalidades que estão sendo desenvolvidas. Essas duas visões de progresso (desempenho e funcionalidade) correspondem à base do monitoramento do progresso de projeto no Inspector.

Capítulo 3

Métricas de Software

Para organizações que realizam desenvolvimento de sistemas, é fundamental a seleção de projetos de acordo com os recursos financeiros e prazos disponíveis. O elemento chave para se fazer isso é o nível de conhecimento adquirido pela organização através da experiência adquirida em projetos anteriores. Fazer comparações e capturar os aspectos positivos dos projetos desenvolvidos pela organização, visando utilizar técnicas de gerenciamento que obtiveram sucesso quando implantadas, reduz em grande proporção os riscos do projeto e aumentam a probabilidade do desenvolvimento do projeto obter o sucesso esperado.

No contexto de acompanhamento de um projeto de software e captura de experiências obtidas durante o desenvolvimento, as métricas desempenham um papel de grande importância, visando fornecer ao gerente uma visão mais completa da situação atual do projeto. Uma métrica corresponde a um padrão de medida usado para julgar os atributos de alguma característica a ser avaliada, tais como qualidade ou complexidade do sistema, de uma maneira imparcial [35]. Usando experiências anteriores como guia, e entendendo como o novo projeto proposto se compara com projetos já completados, a organização tem muito mais chance de terminar o projeto no tempo e dentro do orçamento previsto.

Uma grande variedade de métricas auxiliam no gerenciamento de projetos. Inicialmente esse capítulo apresenta algumas propriedades desejáveis a uma métrica de software e a teoria utilizada para validar mais formalmente a métrica, em seguida apresenta a categoria de métricas tradicionais, as métricas orientadas a objetos e as

métricas de avaliação de progresso, mostra também a visão de métrica segundo diversos autores e, por fim, faz algumas considerações sobre métricas de software.

3.1 Propriedades Desejáveis

Muitos autores têm proposto propriedades desejáveis de métricas de software [9, 12, 20, 23]. Uma métrica deve ser objetiva, ou seja, o valor resultante de seu cálculo deve indicar claramente a situação da característica que ela pretende representar. Ela deve ser intuitiva e possuir alguma forma de ser validada. Além disso, uma métrica deve ser robusta e confiável, visando fornecer um *feedback* ao usuário da métrica. A métrica deve fornecer informação, através de análise estatística que permita ao usuário obter um melhor entendimento do projeto e de como ele pode ser melhorado.

Brito e Abreu em [9] sugerem que os seguintes critérios devem ser considerados na adoção de um conjunto de métricas:

1. Métricas devem ser bem definidas.
2. Métricas devem ser dimensionáveis ou expressas em alguma unidade.
3. Métricas devem ser obtidas o mais cedo possível no ciclo de vida do sistema.
4. Métricas devem ser facilmente calculadas.
5. Métricas devem estar em uma escala que aumente sua precisão.
6. Métricas devem ser vistas como probabilidades de forma a permitir a aplicação de teorias estatísticas sobre as mesmas.

Já Gilb [20] observa que medidas de um processo de desenvolvimento de software devem possuir as seguintes características:

1. As métricas devem ser robustas, ou seja, devem ser precisas e relativamente insensíveis a pequenas mudanças em ferramentas, métodos ou características do produto.
2. As métricas devem sugerir uma norma, indicando os valores ótimos e os valores a serem evitados.

3. As métricas devem relacionar propriedades do produto ou processo, ou seja, devem relacionar o número de erros ao tamanho, aos recursos gastos, etc.
4. As métricas devem sugerir uma estratégia de melhoria.
5. As métricas devem ser resultados naturais de um processo, ou seja, as atividades necessárias para coleta e avaliação das métricas não devem prejudicar o desenvolvimento do produto.
6. As métricas devem ser simples.
7. As métricas devem ser intuitivas e rastreáveis.

Portanto, resumindo, para que uma métrica seja realmente útil e de qualidade, ela deve ser válida, confiável e prática [23]. Diz-se que uma métrica é válida se ela quantifica o que nós queremos medir. Ela é confiável se a aplicação correta do seu algoritmo produz o mesmo resultado dadas as mesmas condições. Além disso, uma métrica é prática quando é barata, fácil de computar e fácil de interpretar. As subseções a seguir descrevem mais detalhadamente cada uma destas propriedades.

3.1.1 Validade

Existem diferentes tipos de validade identificadas na literatura, classificados em três categorias: superficial, interna e externa [23].

Validade superficial não comprova a validade de uma métrica, pois apenas indica que a métrica é intuitivamente válida. Tal validade é útil somente sob uma perspectiva prática. Se uma métrica não possui tal validade, ela dificilmente se aplica corretamente ao projeto. A grande maioria das métricas conhecidas e utilizadas apresentam validade superficial. Por exemplo, a métrica número de classes indica intuitivamente o tamanho do sistema, pois sistemas que possuem mais classes que outros são considerados, intuitivamente, maiores.

Validade interna lida com quão bem uma métrica captura diferenças reais nos valores de um atributo da entidade a ser medida. Por exemplo, usando a métrica número de linhas de código para comparar o tamanho de um programa em C com outro, se o número de vírgulas de um programa é maior que no outro, então ele possui tamanho

maior que o outro. Esta métrica possui validade interna se ela realmente capturar o significado de tamanho. Validade interna pode ser verificada através de experimentos cuidadosamente construídos.

A forma geral de se identificar a validade interna de uma métrica é observar a validade de: conteúdo, critérios relacionados e construção [23]. Validade de conteúdo verifica se a cobertura da métrica sobre o atributo a ser mensurado é boa. Validade dos critérios relacionados verifica se a métrica prediz a ocorrência de um evento futuro ou estima a condição corrente. Validade de construção lida com a identificação dos atributos da entidade, as diferentes métricas dos atributos e as teorias nas quais os atributos foram baseados.

Validade externa endereça a generalidade da métrica. Primeiro verifica se a métrica pode ser generalizada além das entidades que estão sendo medidas atualmente, e em seguida verifica se a métrica pode ser generalizada para ambientes diferentes do que está sendo atualmente usado [23].

3.1.2 Confiabilidade

É uma propriedade necessária à métrica, mas não suficiente. Ela endereça se uma métrica produz ou não resultados consistentes. Basicamente consiste em verificar duas propriedades: estabilidade e equivalência.

Estabilidade lida com a idéia que a métrica deve produzir os mesmos resultados dada uma mesma entidade em um mesmo ambiente. Dependendo do tipo de métrica a ser testada, dois problemas geralmente acontecem. Primeiro, as pessoas podem aprender como a métrica é calculada e modificar a entidade somente para produzir os resultados desejados, sem se preocupar com o processo de desenvolvimento estabelecido. Além disso, o processo de medição utilizado pode alterar o resultado da entidade que está sendo medida, causando efeitos colaterais não desejados.

A propriedade de equivalência verifica o impacto de diferentes amostras ou investigadores na métrica. Por exemplo, é possível verificar se dois programas escritos em uma mesma linguagem são equivalentes em tamanho, se os dois programas

apresentam os mesmos valores para a métrica de tamanho utilizada (linha de código, número de vírgulas, etc.).

3.1.3 Praticidade

De acordo com Card e Glass [12], uma métrica é prática quando ela é econômica para se coletar, é fácil de se entender e bastante informativa. Além disso, as informações que ela resulta devem ser geradas periodicamente para que o *feedback* que ela fornece seja útil.

Métricas para serem realmente úteis devem fazer parte de uma estratégia completa de melhoria no processo de desenvolvimento de software. Além disso, elas devem ser passíveis de serem automatizadas para que os dados relevantes para o cálculo da métrica possam ser mais rapidamente e facilmente coletados e processados [21]. É importante também que a métrica seja independente de linguagem, para que ela possa ser bem aplicada o quanto antes no processo de desenvolvimento de software. A métrica que indica o número de linhas de código, por exemplo, apesar de ser fácil de coletar e avaliar, não pode ser aplicada desde o início do projeto de desenvolvimento, pois é dependente da linguagem de programação, tendo utilidade somente a partir da implementação, o que representa uma falha na propriedade de praticidade.

3.2 Teoria da Mensuração

Uma grande quantidade de métricas vem sendo proposta a cada dia, e não existem padrões ou regras que devem ser obedecidos como forma de garantir a qualidade e a cobertura de uma determinada métrica. Muitas dessas métricas podem apresentar inconsistências durante sua aplicação devido a problemas relacionados com ordem, escala, incompatibilidade de valores, entre outros.

Como já foi dito, a necessidade de medir e controlar o desenvolvimento de software consiste em um fator decisivo para o sucesso de um determinado projeto. Como não existem métricas padrões universalmente testadas e validadas, especialmente para ambientes de desenvolvimento orientado a objetos, o que normalmente se utiliza são

métricas empíricas, ou seja, métricas definidas a partir da experiência em projetos anteriores, que muitas vezes não são adequadas, não representam o que realmente se quer medir ou não apresentam os resultados esperados.

Como consequência da observação desses problemas, tem sido realizado nos últimos anos um grande esforço em fornecer uma base teórica relacionada a métricas, de modo a suportar o desenvolvimento e teste de uma métrica, e garantir a eficiência da mesma. A definição de uma teoria de mensuração identifica se uma métrica específica é apropriada em uma determinada situação ou para um determinado propósito. A Teoria de Mensuração descrita em Zuze [56], Fenton [18] e Baker et.al [3], envolve descrições matemáticas de escalas, medidas, métodos de medição, ordem, significados, de modo a fornecer subsídios teóricos para validar qualquer conjunto de métricas, especialmente nesse trabalho, validar as métricas OO propostas. As subseções seguintes apresentam as características necessárias a uma métrica de acordo com a teoria proposta em [3, 18, 54].

3.2.1 Sistemas Relacionais

A propriedade sobre sistemas relacionais identifica que, se existe um relacionamento intuitivo ou empírico entre os objetos no domínio do problema, estes relacionamentos podem ser formalizados em um sistema relacional formal. Assim sendo, uma métrica representa uma função que faz o mapeamento dos relacionamentos empíricos entre os elementos do projeto em relações formais que representam as medições sobre o projeto.

Seja **A** uma representação empírica do sistema relacional, temos que **A** = (A, R_i) é um par onde A é um conjunto de elementos de projeto do sistema e R_i são as relações empíricas entre estes elementos, tais como “maior”, “maior ou igual”. Nesse estágio não há métricas, as relações de ordem, tamanho e complexidade são observadas intuitivamente. O papel de uma métrica μ é realizar o mapeamento do sistema relacional empírico **A** para um sistema relacional formal **B**, que fornece as relações formais desejadas através de medições. Desse modo **B** = (B, S_i) representa o sistema relacional formal onde B é o conjunto de objetos formais (números, vetores) e S_i são as relações entre os objetos de B [18].

A definição formal da função de mapeamento, a métrica μ, é dada como um homomorfismo[3] $\mu : A \rightarrow B$, que contém o valor $\mu(a)$ em B para todos objetos $a \in A$, ou seja, $\forall a \in A \; \exists b \in B \mid \mu(a) = b$.

3.2.2 Ordem

A ordenação representa outro conceito fundamental da Teoria de Mensuração. Métricas de software devem, preferencialmente, apresentar a noção de ordem que permita a comparação entre valores e avaliação dos mesmos. Basicamente, uma métrica de software deve representar elementos através de uma ordem parcial. Esta ordem apresenta as propriedades reflexiva (aRa, para todo $a \in A$), transitiva (aRb e $bRc \rightarrow aRc$, para todo $a,b,c \in A$) e anti-simétrica (aRb e $bRa \rightarrow a = b$, para todo $a,b \in A$).

3.2.3 Tipos de Escala

Existem diversos tipos de escala para representar os resultados obtidos pela coleta das métricas. Diferenças entre escalas existem porque, para cada escala, existe um número limitado de transformações admissíveis aplicáveis a mesma. Uma transformação admissível é definida como sendo o mapeamento $g : \mu(A) \rightarrow B$ enquanto (A, B, $g \circ \mu$) é uma escala. Outra maneira de dizer isso é que a função $g \circ \mu$ de A para B representa um homomorfismo. Por exemplo, se $\mu(x) = 2x$, então $\mu(x)$ é um homomorfismo, desde que para x_1 e x_2, com $x_1 > x_2$, temos que $\mu(x_1) > \mu(x_2)$, sendo $g : \mu(a) \rightarrow B$ uma transformação de escala admissível. Entretanto, para $g(x) = -x$, $g \circ \mu$ não é um homomorfismo, então, para esse caso, $g : \mu(A) \rightarrow B$ não é uma transformação admissível. O tipo de transformação admissível define o tipo de escala. Os tipos de escalas existentes são [23]:

---

[3] Homomorfismo. S. m. **1.** Qualidade de homomorfo. **2.** *Álg. Mod.* Transformação unívoca de um grupo sobre outro que preserva as operações dos grupos.

- Escalas Nominais: são aquelas cujos rótulos são ligados a objetos, mas nenhum tipo de ordenação é possível ou significante. Permite apenas a utilização de estatísticas não paramétricas, como frequências, médias, etc.

- Escalas Ordinais: são mais "fortes" que as escalas nominais pois permitem ordenação. Estatísticas de ordem são possíveis além daquelas permitidas em uma escala nominal, entretanto, operações aritméticas não são significativas para métricas com esse tipo de escala. Uma outra maneira de ver uma escala ordinal, é uma escala onde as relações são definidas, mas não é possível aplicar operações binárias sobre ela.

- Escalas de Intervalo: além de permitir a ordenação dos objetos, temos que a distância entre um par de objetos também tem significado, por exemplo a temperatura medida em graus Celsius. Dessa forma, as transformações admissíveis nessa escala devem preservar tanto a noção de ordem quanto a de intervalo. Qualquer transformação linear do tipo $g(x) = ax + b \ (a > 0)$ faz isso.

- Escalas de Taxa: permite cálculos mais complexos, baseadas em taxas e percentuais. As transformações admissíveis para esse tipo de escala são do tipo $g(x) = ax \ (a > 0)$. Exemplos desse tipo de escala são a massa, a temperatura em Kelvin, o comprimento e o intervalo de tempo. Nesse tipo de escala, o zero tem um valor significativo.

- Escalas Absolutas: apresenta uma grande restrição com respeito as transformações admissíveis que devem ser da forma $g(x) = x$, isto é, a função identidade. Em outras palavras, existe um único valor admissível que é derivado de algum objeto. Desse modo, escalas absolutas representam uma contagem e permitem uma grande variedade de estatísticas descritivas a serem aplicadas, por exemplo, significados e desvios padrões. Um exemplo seria a métrica linhas de código (LOC, *lines of code*), que pode se perceber, está sobre uma escala absoluta, onde a contagem verifica o número de linhas de código.

Diversos autores sugerem que as métricas devem estar fundamentadas em uma escala de taxa, pois ela permite realizar médias significativas e verificar percentuais. Zuse [56] indica que escalas de intervalo não são apropriadas para métricas, enquanto Harrison et al. [22] sugerem que a melhor escolha é a escala ordinal. Para se conseguir uma transformação com escala do tipo taxa, Zuse [56] argumenta que é necessária uma estrutura extensiva, que permita adicionar operações binárias. Outra vantagem de escalas de taxa é que o valor zero possui significado. Métricas como complexidade ainda apresentam dificuldades em indicar o significado do valor zero, entretanto outras como intervalo de tempo definem claramente o significado desse valor.

3.2.4 Estruturas Extensivas

Outra propriedade relevante a uma métrica é garantir que ela representa uma estrutura extensiva. Uma estrutura extensiva representa um veículo formal para adicionar operações binárias à métrica, tais operações binárias são representadas através de $\circ$ em um sistema empírico. Seja um conjunto P com uma relação binária $\bullet\geq$, e uma operação binária $\circ$, então a estrutura relacional (P, $\bullet\geq$, $\circ$) é uma estrutura extensiva se e somente se $\forall p1...p4 \in P$:

1. (P, $\bullet\geq$) representa uma relação de fraca ordem ($\geq, \leq$)
2. $p_1 \circ (p_2 \circ p_3) \approx (p_1 \circ p_2) \circ p_3$ (fraca associatividade)
3. $p_1 \circ p_2 \approx p_2 \circ p_1$ (fraca comutatividade)
4. $p_1 \bullet > p_2 \Leftrightarrow p_1 \circ p_3 \bullet > p_2 \circ p_3 \Leftrightarrow p_3 \circ p_1 \bullet > p_3 \circ p_2$ (fraca monotonicidade)
5. Se $p_3 \bullet \geq p_4, \forall p_1, p_2 \exists$ número natural n, tal que $p_1 \circ np_3 \bullet \geq p_2 \circ np_4$ (axioma de Arquimedes)

3.2.5 Atomicidade

Uma modificação atômica representa uma mudança única em uma lista de mudanças, ou seja, uma mudança que não pode ser subdividida em mais mudanças no projeto. Para a métrica LOC (linhas de código), por exemplo, modificações atômicas poderiam ser a inclusão, remoção ou transferência de uma linha de código. A sensibilidade da métrica em relação a essas mudanças atômicas indica as propriedades

parciais da métrica, ou seja, quais fatores influenciam a métrica [23]. Um exemplo seria, para a métrica LOC, realizar as modificações:

M1: adicionar uma linha de código no método m1 e remover outra linha no método m2.
M2: transferir uma linha de um local para outro.
M3: Adicionar uma nova linha.

O que se observa é que as modificações M1 e M2 não resultam em mudança no valor de LOC, enquanto M3 aumenta seu valor. Como o objetivo da métrica LOC é indicar o tamanho do sistema em relação ao número de linhas de código existente, temos que as modificações atuais não invalidam as relações empíricas inicialmente pretendidas. Se o resultado de alguma modificação atômica não traduz em um comando aceitável dentro do sistema relacional empírico definido pela métrica, temos que a métrica deve ser rejeitada.

3.3 Métricas de Processo e Métricas de Produto

Para conseguir visualizar como as métricas podem ser úteis para organização, é interessante agrupar as características fundamentais de sistemas genéricos em duas categorias distintas: métricas de processo e métricas de produto. Tais categorias são conhecidas como tradicionais [51], e se distinguem no objetivo da avaliação, onde métricas de processo se preocupam com os acontecimentos ocorridos entre a última avaliação e a avaliação anterior, e métricas de produto se preocupam com a avaliação do produto em um momento particular.

3.3.1 Métricas de Processo

São conhecidas também como métricas de gerenciamento, relacionam-se ao processo usado para construir o sistema [23]. Geralmente métricas de processo são usadas pelos desenvolvedores para:

- ajudar a fazer uma previsão do tamanho do sistema final;
- derivar o nível de esforço que um projeto irá precisar; e

- determinar se o projeto está em dia com o plano de trabalho previamente determinado.

Métricas de processo não se preocupam com a qualidade do sistema em si, apesar de que, projetos com métricas de processo bem definidas e acompanhadas tendem a resultar em sistemas com qualidade. A função destas métricas é avaliar o tamanho e o estado do sistema, servindo como um referencial para acompanhamento da eficiência na qual o processo está sendo implantado. Desta forma, métricas de processo descrevem seu projeto sem prescrever ações corretivas para o produto. Apesar disso, de uma perspectiva geral, métricas de processo são fundamentais e mais necessárias do que métricas de produto [51].

3.3.2 Métricas de Produto

Tais métricas, conhecidas também como métricas de qualidade, têm como objetivo auxiliar a medir a qualidade dos sistemas. Henderson-Sellers [23], descreve um número de categorias para avaliação da qualidade de um sistema: confiabilidade, disponibilidade, manutenabilidade, entendibilidade, modificabilidade, testabilidade e usabilidade. Geralmente métricas de produto são usadas para fornecer:

- diretrizes que sugerem ações locais e específicas para melhorar a qualidade de diferentes componentes do sistema;
- comparações entre sistemas existentes; e
- comparações entre sistemas novos e outros sistemas conhecidos.

Não existe uma relação bem definida entre métricas de produto e métricas de processo. A qualidade do sistema é um aspecto crítico que deve ser levado em consideração para resultar em um sistema com boa funcionalidade e usabilidade. Métricas de qualidade fornecem discernimento entre as diferentes propriedades observáveis de um sistema, para que a qualidade prevista para o sistema seja atingida.

3.3.3 Aplicando as Métricas

O gerente de um determinado projeto de software tem uma variedade de usos para métricas de software. Apesar das diferenças entre métricas de processo e de produto, geralmente usa-se ambas, em conjunção, para avaliar o projeto como um todo. Mais especificamente, métricas são utilizadas para prover algum tipo de informação quantitativa, visando auxiliar o gerente a adotar decisões relacionadas a três áreas do ciclo de vida do projeto:

- definição do custo e tempo necessário;
- determinação do estado do projeto e do esforço necessário para completá-lo; e
- qualidade do produto.

Propor uma estimativa correta e precisa para o projeto a ser realizado é fator crítico para o sucesso do projeto, ou seja, é de fundamental importância a definição realística dos custos financeiros e de tempo. Algumas vezes, a definição dos custos necessários pode indicar ao gerente a inviabilidade da realização do projeto. Além disso, estimativas precisas podem também ajudar o gerente a fazer um balanceamento entre a qualidade do produto final, e os custos de tempo e financeiro previstos.

Métricas oferecem uma visão excelente do estado corrente do projeto e um bom entendimento do esforço que ainda deverá ser realizado para completar o mesmo. Através delas o gerente de projeto pode identificar os problemas ocorridos em áreas específicas, tornando mais fácil fazer ajustes no escopo, no cronograma e no plano de trabalho.

Durante todo o projeto, métricas de qualidade auxiliam a guiar a produção de sistemas robustos e de alta qualidade. Produtos de baixa qualidade podem, algumas vezes, gerar dificuldades de aceitação por parte do cliente, ou ainda resultar em grandes gastos para manutenção do sistema. Métricas bem definidas podem salientar propriedades de qualidade importantes e fornecer ações corretivas antes do problema tomar maior proporção.

3.4 Métricas Orientadas a Objetos

A classificação de métricas em métricas de processo e métricas de produto é a mais tradicional e considera a construção de softwares genéricos (independente do paradigma de desenvolvimento, linguagem de programação, etc.). O que se pode perceber é que os resultados de uma métrica variam muito dependendo do processo e do paradigma utilizado para o desenvolvimento [14]. Nesse contexto, verifica-se a necessidade de classificar as métricas de software mais adequadas ao paradigma orientado a objetos (OO). As métricas de desenvolvimento de software OO, podem ser classificadas em quatro categorias, segundo Schroeder [51]:

- **Tamanho do Sistema:** preocupa-se com o tamanho e a complexidade do sistema. Por exemplo, quantas chamadas estáticas a funções e objetos o sistema realiza.
- **Tamanho de Classe ou Método:** ainda que classes e métodos possam ser medidos e caracterizados de várias maneiras, classes ou métodos pequenos e simples são tipicamente melhores projetados do que outros (com a mesma funcionalidade) maiores e mais complexos. O acompanhamento dessa categoria resulta em sistemas com melhor manutenabilidade e usabilidade.
- **Acoplamento e Herança:** o número e o tipo destes relacionamentos indicam a interdependência entre classes. Claramente, relações simples são preferíveis a relações numerosas e complexas.
- **Classes ou Métodos Internos:** esta métrica revela a complexidade interna das classes e métodos, e verificam o quão bem documentado está o código do sistema.

Lamentavelmente, métricas de tamanho do sistema não possuem valores padrões que possam ser comparados para avaliar o sistema que está sendo construído. O tamanho do sistema depende inteiramente da quantidade de funcionalidade que está inserida dentro do mesmo. Outras métricas, no entanto, têm valores padrões. Por exemplo, o tamanho de um método é razoavelmente consistente entre os diversos sistemas.

Para métricas OO que não sejam o tamanho do sistema, faz sentido falar sobre médias e cálculos estatísticos a nível de sistema. Por exemplo, é indicado perguntar se os métodos do sistema estão, na média, maiores do que aqueles de sistemas semelhantes anteriormente desenvolvidos. Tais médias fornecem uma indicação da qualidade e do progresso do sistema como um todo, mostrando sinais que podem afetar o desenvolvimento do sistema. A seguir serão apresentadas, mais especificamente, cada uma das métricas OO anteriormente citadas.

3.4.1 Tamanho do Sistema

Métricas relativas ao tamanho do sistema estão relacionadas ao esforço total requerido para a construção do sistema. Na maioria dos sistemas OO, componentes da interface gráfica (GUI) representam uma quantidade significativa do trabalho de desenvolvimento, juntamente com outras métricas relativas ao tamanho do sistema [51]. Algumas das métricas para avaliação do tamanho de um sistema são [30, 35, 44, 51]:

- **Total de Linhas de Código:** esta métrica, também conhecida como LOC (*lines of code*), conta todas as linhas de código executável, ignorando comentários, no sistema, em uma classe ou método. Desenvolvedores usam esta métrica como uma medida do tamanho do sistema e, desde que o número de linhas de código pode ser contabilizado automaticamente, pode-se utilizar ela como uma forma de comparação com diferentes sistemas. Maiores detalhes sobre tal métrica podem ser encontrados em [44].
- **Total de Chamadas a Funções:** esta métrica, também conhecida como TFC (*total function calls*), conta o número de chamadas a métodos e funções dentro do sistema, de uma classe ou método. Esta métrica mede o tamanho de uma maneira mais independente do estilo de codificação adotado na organização do que a métrica LOC, mas ainda assim depende muito do paradigma de desenvolvimento e do tipo de linguagem de programação que está sendo usado.
- **Número de Classes:** esta métrica, também conhecida como NOC (*number of classes*), conta todas as classes dentro do sistema, incluindo classes não visuais, e classes de GUI, como janelas, combo-box, etc.

- **Número de Subsistemas:** esta métrica mede o tamanho do sistema em relação aos diversos subsistemas que ele possui. Verifica as coleções de classes que suportam um conjunto de funções de usuários finais.
- **Número de Janelas:** esta métrica, também conhecida como NOW (*number of windows*), conta o número de janelas visíveis dentro do sistema. Este número indica o tamanho da interface do usuário.
- **Número de Cenários:** conta o número de cenários que são usados para documentar o nível de usabilidade do sistema. Relacionada com as responsabilidades públicas dos subsistemas e classes a serem desenvolvidos [30].

3.4.2 Tamanho de Classe e Método

Na realidade, pode-se considerar métricas de tamanho de classes e métodos como sendo métricas de qualidade, porque classes ou métodos muito grandes podem indicar abstrações mal realizadas, ou ainda implementações excessivamente complexas [51]. Medidas do tamanho de classes e métodos que se diferem substancialmente de valores médios adquiridos de experiências passadas são geralmente bons candidatos para inspeção ou um novo desenvolvimento. Entre as técnicas que podem ser usadas para medir o tamanho de uma classe e/ou um método estão [35, 51]:

- **LOC e Chamadas de Funções por Classe/Método:** estas métricas são similares as métricas usadas para medir o tamanho do sistema, mas focaliza as classes e os métodos separadamente.
- **Número de Comandos por Método:** esta métrica indica o tamanho do método de acordo com o número de comandos que o mesmo realiza quando invocado.
- **Número de Métodos por Classe e o Número de Métodos Públicos por Classe:** o número de métodos por classe indica o nível de funcionalidade implementada por uma classe. O número de métodos públicos indica a quantidade de comportamento exposto para o mundo externo, fornece uma visão do tamanho da classe e como a classe poderia ser utilizada por outras.

- **Número de Atributos e Número de Instâncias de Atributos por Classe:** o número de atributos em uma classe indica a quantidade de dados que a classe deve manter para executar o que foi proposto. Atributos podem ser: instância de atributos, que são únicos para cada instância de um objeto, ou variáveis de classe, que tem o mesmo valor para todos os membros da classe.

3.4.3 Acoplamento e Herança

Tais métricas auxiliam na medição da qualidade de um modelo de objetos. Na realidade esta métrica ajuda a revelar o grau de dependência entre os objetos existentes [51]. Idealmente, objetos deveriam ser o mais independente quanto possível, pois isso tornaria mais fácil transportar um objeto de um ambiente para outro e reusar objetos existentes quando se fossem construir novos sistemas. A realidade, no entanto, é bastante diferente do ideal, geralmente objetos são dependentes de outros, e reusá-los raramente é uma simples operação de colagem. Normalmente, diversas modificações devem ser realizadas na classe a ser reusada, além disso, classes que não são interessantes ao novo sistema podem ser inseridas para conseguir adaptar a classe do sistema antigo para o sistema novo. Geralmente o número de dependências é tão grande que entender e mover um grupo de objetos para o novo sistema é mais complicado do que reescrever os objetos partindo do início [51].

- **Nível de Convergência a uma Classe (*fan-in*):** esta métrica mede o número de classes que dependem de um dado objeto. Interessante para aqueles que desejam acoplar os objetos, pois ela centraliza as dependências sobre um único objeto, como ilustrado na Figura 3.1, onde cada seta representa a dependência de uma classe à outra, nesse caso, as classes A, B, C e D dependem da classe E. Usando esse tipo de técnica, pode-se reusar qualquer uma das classes relacionadas em outro projeto simplesmente incluindo a classe E.

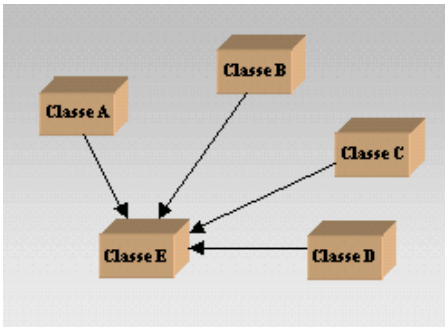


Figura 3.1. Nível de convergência de uma classe

- **Espalhamento de uma Classe (*fan-out*):** esta métrica mede o número de classes sobre as quais uma certa classe depende. Não é interessante que sistemas contenham esta propriedade, pois ela representa uma situação na qual a dependência entre objetos fica espalhada, ou seja, não fica centralizada. A Figura 3.2 ilustra este espalhamento.

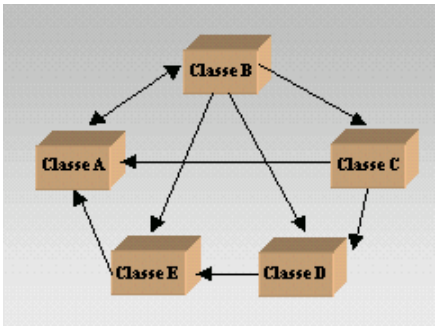


Figura 3.2. Espalhamento de uma classe

- **Profundidade de Herança de uma Classe:** a profundidade de herança de uma classe corresponde ao nível da subclasse mais profunda (folha) na árvore de herança da classe. Uma profundidade de herança desnecessária aumenta a complexidade e pode representar um mau uso do mecanismo de herança.
- **Número de Filhos por Classe:** esta métrica mede o número de descendentes diretos de uma classe particular, que pode indicar a existência desnecessária de uma hierarquia complexa.
- **Número de Classes Abstratas:** esta métrica é uma indicação do sucesso no uso de herança e o esforço que tem sido gasto observando conceitos gerais no domínio do problema.

- **Número de Métodos Adicionados pela Subclasse:** subclasses devem definir novos métodos, estendendo o comportamento das superclasses. Uma classe sem método é geralmente questionável.
- **Número de Métodos Herdados pela Subclasse:** indica a força da especialização e se a herança é realmente justificável. Subclasses naturalmente herdam o comportamento da superclasse em forma de métodos, e os dados estáticos em forma de variáveis.

3.4.4 Classes e Métodos Internos

Métricas internas fornecem várias medidas de qualidade, tanto para classes quanto para métodos. Elas indicam as áreas onde ações corretivas podem ser úteis. A nível de sistema, tais métricas também podem ser usadas para comparar a qualidade de um sistema com outro. Entre essas métricas estão inclusas [35, 51]:

- **Número de Referências Globais/Compartilhadas por Classe:** esta métrica indica o número de referências a variáveis globais encontradas dentro de uma classe. Referências globais tendem a interromper o encapsulamento e inibir o reuso. Eliminar completamente as referências globais é tarefa bastante difícil, nesse caso, tais referências deveriam ser, pelo menos, usadas com moderação.
- **Complexidade do Método:** tal métrica mede a complexidade a nível de ciclo, ou seja, o número de caminhos de execução diferentes que podem ocorrer dentro de um bloco de código. Código com muitos caminhos são difíceis de se entender e mais comuns de apresentarem erros.
- **Número de Atributos Públicos por Classe:** tal métrica conta todos atributos que não estejam marcados como protegidos ou privados. Estes atributos podem expor a implementação de um objeto para o mundo exterior, o que viola os princípios para um bom encapsulamento do objeto.
- **Falta de Coesão entre Métodos:** esta métrica mede o quanto métodos referenciam dados da instância da classe. Em geral, projetos de alta qualidade têm baixo acoplamento (interação entre objetos) e alta coesão (objetos que não podem ser repartidos/divididos).

- **Índice de Especialização de Classes:** esta métrica verifica o quanto subclasses sobrepõem (*override*) o comportamento das suas classes ancestrais. Uma quantidade excessiva de *overridden* de métodos indica que a abstração pode ter sido realizada de forma inadequada, desde que, nesse caso, a classe filha e seus ancestrais podem não ter muito em comum.
- **Número de Métodos Públicos em uma Classe:** é uma boa métrica para verificar a quantidade de responsabilidade imposta a uma classe. Os métodos públicos compreendem os contratos testados durante a verificação e dirige o trabalho feito na classe.
- **Porcentagem de Métodos Comentados:** tal métrica mede a quantidade de código que está internamente documentada. Ela conta o número de blocos de comentário separados, ao invés de contar o número de linhas de comentário.
- **Número de Parâmetros por Método:** esta métrica encontra o número médio de parâmetros envolvidos na invocação de um método. Um número de parâmetros excessivo indica uma interface complexa para chamada de objetos e, portanto, deve ser evitada.

3.5 Métricas de Avaliação de Progresso

A avaliação de progresso consiste no foco principal deste trabalho, que visa definir um processo para avaliar progresso de projetos de software OO. Nesse contexto, é importante definir e apresentar, como forma de apoio à pesquisa, alguns conceitos e métricas relacionados com a avaliação de progresso.

A primeira questão a responder é: o que é progresso? Fala-se muito em progresso no desenvolvimento de software, mas, muitas vezes, passa-se desapercebido que a noção de progresso não é bem definida. O progresso de um determinado projeto de software corresponde a um indicador[4], que fornece informação sobre quão bem o projeto está sendo realizado, com respeito ao cronograma de atividades e aos serviços

[4] Um indicador é uma propriedade utilizada para monitorar algum aspecto do projeto.

que devem ser fornecidos para o cliente [5]. Um indicador é quantificado através de métricas, que são calculadas e resultam em valores que permitem obter uma avaliação precisa do indicador.

O progresso é avaliado, verificando as atividades completas no cronograma do projeto. Os indicadores de progresso são usados para monitorar o progresso, em termos de tarefas que foram concluídas e das saídas produzidas por estas tarefas. A diferença entre a situação atual (atividades que foram realizadas) e o planejado (atividades que deviam ter sido realizadas), é uma indicação da aderência do projeto ao plano. Desvios significativos em relação ao esperado indicam problemas. Além disso, outra grande utilidade dos indicadores de progresso é mostrar tendências, que podem apresentar a possibilidade de surgirem problemas futuros.

Métricas de progresso podem ser usadas para monitorar as atividades durante todo o ciclo de vida de desenvolvimento do software. Cada etapa do ciclo de vida tem atividades que devem ser realizadas, e que podem ser quantitativamente medidas com métricas de processo. E ainda, cada atividade produz saídas tangíveis, tais como código fonte ou projeto de algum subsistema, que podem ser efetivamente monitorados com métricas de produto. Tais métricas fornecem ao gerente informações, não somente sobre o produto, mas também sobre o processo utilizado para desenvolver o produto.

Avaliação de progresso pode ser usada por todos os níveis de gerenciamento de projeto. Os gerentes preparam o cronograma de atividades, que inclui as datas esperadas para início e fim de cada atividade, e recuperam as datas reais em que estas atividades foram realizadas. De posse dessas informações, os gerentes monitoram o progresso de acordo com o que fora anteriormente planejado, cada gerente verificando sua área de controle. Monitorar uma determinada atividade, consiste em verificar o desvio do progresso atual do projeto em relação ao esperado, além de observar tendências indicadas pela taxa de progresso.

Durante a avaliação de progresso, diversos artefatos são capturados como entrada, e a partir da avaliação de cada artefato é que pode ser verificado o progresso global do projeto. Alguns destes artefatos são:

- Requisitos analisados;
- Requisitos do software documentados;
- Casos de Teste especificados;
- Elementos de Projeto concluídos;
- Unidades implementadas e testadas; e
- Casos de Teste executados.

A melhor forma de visualização das atividades do plano de desenvolvimento de software são gráficos, que indicam de forma clara o *status* atual de cada atividade, e permitem fácil comparação com os valores esperados [23]. Gráficos *Gantt* constituem em um dos mais tradicionais gráficos de auxílio ao acompanhamento de projetos, podendo ser criado a partir das principais ferramentas de planejamento do mercado (por exemplo, o Microsoft Project [45]). Consistem em gráficos de responsabilidades que indicam, para cada atividade, a data de início e fim, além de conter comentários sobre a mesma. A Figura 3.3 apresenta um exemplo de gráfico Gantt.

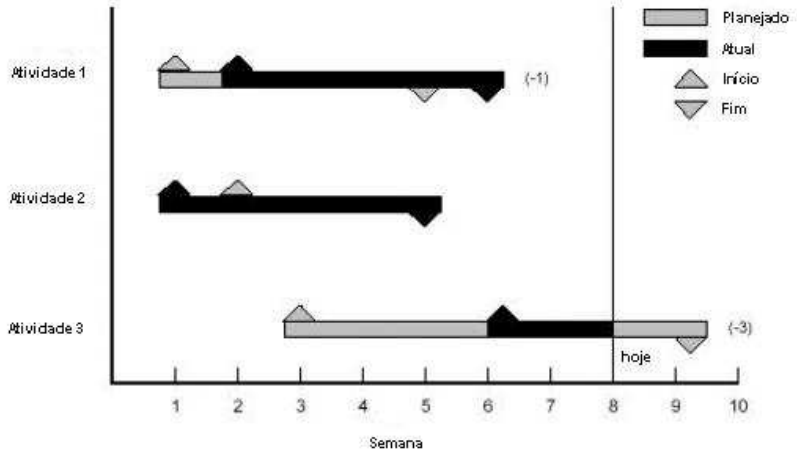


Figura 3.3. Gráfico Gantt para planejamento e acompanhamento

De posse de um gráfico *Gantt*, o gerente pode observar cada atividade inerente ao projeto, identificando se as mesmas já foram concluídas ou não. Caso tenham sido realizadas, ele verifica se as datas estão de acordo com o planejado, ou se há a necessidade de realizar deslocamentos no cronograma, a fim de adequar as datas à produtividade obtida durante o desenvolvimento.

Além do acompanhamento das atividades realizadas durante o processo de desenvolvimento, o tamanho do sistema também se relaciona ao progresso do sistema. É importante considerar, em cada relatório periódico de avaliação, o tamanho atual do sistema em relação ao tamanho esperado. As métricas de avaliação de progresso podem ser classificadas em três categorias distintas, de acordo com o foco da avaliação. São elas: desempenho, progresso das unidades de trabalho e capacidade incremental [5].

3.5.1 Desempenho

As métricas de desempenho fornecem informações básicas sobre o progresso no cronograma de atividades e eventos de um determinado projeto. Tais métricas, também ajudam a identificar e avaliar dependências entre atividades e eventos, relativas ao desenvolvimento de software [30]. Monitorar mudanças no cronograma inicialmente planejado, permite ao gerente de projeto avaliar os riscos relacionados com a realização dos futuros marcos de referência[5]. Algumas das principais métricas utilizadas para avaliação do desempenho do projeto são:

- **Número de Iterações:** somente para o caso de processo de desenvolvimento iterativo[6]. Uma iteração é um esforço no desenvolvimento de uma série de itens. Tal métrica identifica o número de vezes em que deverá ser desprendido esforço para desenvolvimento de funcionalidades do sistema. Iterações permitem realizar uma validação prévia do esforço gasto no desenvolvimento.
- **Número de Contratos Concluídos:** verifica quanto dos serviços públicos, ou seja, serviços que o sistema deve possuir para satisfação do cliente, foram completamente desenvolvidos [35]. Geralmente uma iteração representa a realização de diversos contratos.

[5] Um marco de referência representa a realização de alguma atividade que indique, claramente, que o projeto obteve progresso no seu desenvolvimento.

[6] Um processo de desenvolvimento iterativo representa o desenvolvimento do sistema através de um conjunto de iterações, onde cada iteração representa um mini-projeto [7].

- **Datas dos Marcos de Referência:** mede o desempenho da equipe de desenvolvimento para a realização dos marcos de referência do projeto. Exemplos de marcos de referência são o documento de requisitos do projeto definido e documentado, ou ainda uma versão do sistema distribuída [7]. Marcos de referência fornecem um fácil entendimento do *status* do cronograma de atividades do projeto.

3.5.2 Progresso das Unidades de Trabalho

Medidas de progresso das unidades de trabalho se preocupam em verificar o progresso, baseado no quão completo estão as unidades de trabalho, que se combinam incrementalmente para formar um produto ou atividade de software completa [30]. Unidades de trabalho correspondem aos produtos que são gerados e documentados durante o desenvolvimento do projeto, por exemplo, componentes de software, requisitos, casos de teste, etc. Se critérios objetivos, que indicam se uma unidade está ou não concluída, forem definidos, tais medidas são extremamente efetivas para avaliar progresso em qualquer ponto do projeto. Elas são usadas para projetar as datas de finalização de uma atividade ou produto. Algumas das principais métricas utilizadas para avaliação das unidades de trabalho do projeto são:

- ***Status* dos Componentes:** conta o número de componentes de software que têm completado uma atividade específica do desenvolvimento. Uma comparação entre os componentes atuais e os esperados é uma maneira efetiva de avaliar o progresso no desenvolvimento.
- ***Status* dos Requisitos:** conta o número de requisitos definidos que têm sido alocados à componentes de software e casos de teste, e aqueles que têm sido testados com sucesso. Indica o progresso no projeto e nos testes do software. Verifica o grau de funcionalidade inserida no sistema, em relação aos requisitos especificados, bem como a quantidade de teste que tem sido realizada.
- ***Status* dos Casos de Teste:** conta o número de casos de teste que têm sido focalizados e completados com sucesso. Avalia a qualidade do software,

baseado na proporção de casos de teste executados com sucesso em relação ao número de testes desenvolvidos.

- **Cenários Testados:** Verifica o número de caminhos lógicos testados com sucesso. Esta métrica mostra a proporção do software que está testada e de acordo com os requisitos do sistema.
- ***Status* do Relatório de Problemas:** conta o número de problemas documentados e solucionados. Esta métrica fornece uma indicação da maturidade do produto, e também pode ser usada como uma indicação da qualidade do processo de resolução de problema.

3.5.3 Capacidade Incremental

Medidas que indicam a capacidade incremental, verificam o incremento funcional no conteúdo do produto associado a cada distribuição (iteração). Uma distribuição incremental pode ser um produto a ser entregue para o cliente, ou um *build*[7] interno distribuído para a próxima fase do desenvolvimento [30]. Estas métricas são usadas para determinar se a capacidade está sendo desenvolvida como planejada, ou está sendo transferida para iterações futuras. Algumas das principais métricas utilizadas para avaliação da capacidade incremental do projeto são:

- **Número de Componentes do *Build*:** identifica os componentes que estão incluídos em *builds* incrementais. Muitas vezes, para se preservar a data de distribuição do produto, alguns componentes planejados para estarem disponíveis no *build* não são desenvolvidos. É mais fácil verificar a incorporação de capacidade através de componentes do que de funcionalidades, basta verificar se um componente está ou não integrado. Entretanto, esta métrica fornece menos informações, pois nem sempre é fácil indicar a correlação entre os componentes e a funcionalidade [30].

[7] Um *build* é um conjunto de componentes que compõem o sistema em uma fase qualquer do desenvolvimento deste.

- **Funcionalidades do *Build*:** identifica a funcionalidade inserida em *builds* incrementais. Esta métrica indica o progresso na incorporação da funcionalidade incremental [30]. Muitas vezes, para se preservar o cronograma de atividades, nem todas as funcionalidades planejadas para um *build* são desenvolvidas.

3.6 Diversos Conjuntos de Métricas

As subseções a seguir apresentam as opiniões e estudos de diversos pesquisadores, a respeito da utilização e classificação de métricas como forma de auxílio ao gerenciamento de projeto.

3.6.1 Métricas segundo Chidamber e Kemerer

Chidamber e Kemerer [14], propõem um conjunto padrão de métricas para projetos de software OO. Eles justificam a definição desse conjunto de métricas, específico para projetos OO, ao fato do paradigma OO e o paradigma estruturado serem diferentes em sua essência e, portanto, cada um deve possuir métricas específicas. As métricas propostas foram retiradas a partir da Teoria de Mensuração [56], e das experiências de vários desenvolvedores em diversos projetos. Abaixo segue uma breve descrição de cada métrica definida:

- **Complexidade da Classe a partir dos Métodos:** esta métrica, também conhecida como WMC (*Weighted Methods per Class*) mede a complexidade de uma classe individual. Se considerarmos todos os métodos de uma classe como sendo igualmente complexos, WMC é o número de métodos definidos em cada classe [4].
- **Profundidade da Árvore de Herança:** esta métrica, também conhecida como DIT (*Depth of Inheritance Tree*), define a profundidade máxima do grafo de herança de uma classe.
- **Número de Filhos:** esta métrica, também conhecida como NOC (*Number of Children*), determina o número de subclasses imediatas subordinadas a classe na hierarquia de classes.

- **Acoplamento entre Objetos de Classes:** esta métrica, também conhecida como CBO (*Coupling Between Objects*), proporciona o número de classes com as quais uma dada classe está acoplada. Duas classes estão acopladas, quando métodos declarados em uma usam métodos ou variáveis de instância definidas pela outra.
- **Resposta para uma Classe:** esta métrica, também conhecida como RFC (*Response For a Class*), determina o número de métodos que podem potencialmente ser executado, em resposta a uma mensagem recebida por um objeto de uma classe.
- **Falta de Coesão em Métodos:** esta métrica, também conhecida como LCOM (Lack of Cohesion in Methods), gera um número que resulta da subtração do número de pares de métodos sem variáveis de instância compartilhadas, menos o número de pares de métodos com variáveis de instância compartilhadas, de uma classe. O desejável é um valor alto para a coesão da classe, isto é, um baixo valor de LCOM, que recebe zero se o valor da subtração resultar negativo.

3.6.2 Métricas segundo Lorenz e Kidd

De acordo com Lorenz e Kidd [35], a importância das métricas se deve ao efeito econômico que a utilização das mesmas representa para o desenvolvimento de um projeto de software. Eles classificam as métricas em duas categorias distintas: métricas de sistema e métricas de projeto.

Métricas de sistema são usadas para predizer as necessidades do sistema, tais como pessoal alocado para uma tarefa e esforço total para a construção do sistema. Também mede as mudanças dinâmicas no estado do projeto, o quanto tem sido feito e o quanto ainda é necessário fazer. Desse modo, são métricas mais globais do que métricas de projeto. Já métricas de projeto medem o estado estático do projeto em um ponto particular no tempo. Estas métricas focalizam algum aspecto do projeto mais intimamente e, portanto, são mais descritivas. Elas se preocupam com a qualidade do sistema que está sendo construído. Dentro de cada uma destas categorias, as métricas são subdivididas em grupos logicamente relacionados, tais como tamanho do método e

aspectos internos da classe. A Tabela 3.1 contém as principais métricas apresentadas por Lorenz e Kidd de acordo com cada categoria.

Métricas de Sistema	Métricas de Projeto
Tamanho da Aplicação Número de *Scripts* de Cenário Número de Classes Chave Número de Classes de Suporte Número de Subsistemas **Tamanho de Pessoal** Pessoas-dia por Classe Classes por Desenvolvedor **Cronograma** Número de Iterações Número de Contratos Completados	**Tamanho do Método** Número de Mensagens enviadas Linhas de Código (LOC) **Aspectos Internos do Método** Complexidade do Método Mensagens de *Strings* Enviadas **Tamanho da Classe** Número de instâncias de métodos Número de instâncias de variáveis Número de Métodos da Classe Número de Variáveis da Classe **Herança da Classe** Profundidade da Herança Herança Múltipla **Herança de Método** Número de *Override* de Métodos Número de Métodos Herdados **Aspectos Internos da Classe** Coesão Uso Global Uso de Instâncias de Variáveis **Aspectos Externos da Classe** Acoplamento Reuso Número de Colaborações **Acoplamento de Subsistema** Relacionamento entre Subsistemas Relacionamento entre Classes

Tabela 3.1. Principais métricas segundo Lorenz e Kidd.

3.6.3 Métricas segundo Wiegers

Wiegers [55], garante que métricas ajudam a controlar seus projetos de software, além de ensinar mais sobre a maneira que a organização trabalha. Ele utiliza um pequeno e balanceado conjunto de métricas, que auxiliam a organização a rastrear o progresso na realização de suas metas. Ele sugere a técnica GQM (Goal Question Metric), como forma de selecionar as métricas apropriadas para verificar suas necessidades e objetivos. No GQM, inicialmente se seleciona as principais metas da organização, a partir disso, as metas são subdivididas, até que ela possa ser verificada de uma maneira quantitativa (através de uma ou mais métricas) [52]. Além disso, ele propõe um conjunto de métricas, classificadas de acordo com os interesses dos

desenvolvedores, das equipes de desenvolvimento e da organização, como mostrado na Tabela 3.2.

Métricas de acordo com o interesse do grupo
Desenvolvedores
Duração/esforço estimado e atual de uma tarefa
Cobertura de código das unidades de teste
Número de defeitos por unidade de teste
Complexidade do código e do projeto
Equipes de Desenvolvimento
Tamanho do Produto
Status dos Requisitos
Casos de teste aprovados
Duração dos Marcos de Referência
Número de defeitos encontrados durante a integração
Estabilidade dos requisitos
Número de tarefas planejadas e realizadas
Organização
Níveis de defeitos por versão
Tempo de desenvolvimento dos produtos
Precisão das estimativas de esforço e tempo
Efetividade do Reuso
Custo atual e planejado

Tabela 3.2. Métricas segundo Wiegers

3.6.4 Métricas segundo Champeaux

Champeaux [13] considera as métricas em um processo de desenvolvimento que utiliza o paradigma OO. Ele classifica uma métrica de acordo com a etapa do desenvolvimento a qual ela melhor se adequa. Tais métricas são úteis, tanto para gerentes de projeto, quanto para desenvolvedores, de forma a permitir que esses desenvolvedores participem mais do gerenciamento de projetos, facilitando o trabalho de coleta e análise das métricas, que geralmente caem nas mãos do gerente.

As métricas são divididas seguindo as seguintes etapas do desenvolvimento OO: análise, projeto e implementação. Cada métrica é aplicada sobre um artefato, ou conjunto de artefatos resultantes das etapas do desenvolvimento e, a partir disso, ele verifica se o projeto apresenta os resultados esperados. Algumas das principais métricas definidas por Champeaux, em cada etapa do desenvolvimento, são mostradas na Tabela 3.3.

Métricas de acordo com a etapa no desenvolvimento
Análise
Número de ações do usuário no sistema
Número de subsistemas
Número de entradas no vocabulário[8]
Profundidade de herança
Número de classes filhas
Falta de coesão
Complexidade do Diagrama de Classe
Número de métodos e atributos de uma classe
Projeto
Tamanho do Produto
Flexibilidade
Modificabilidade
Mudanças internas nos objetos
Manutenção do sistema
Implementação
Tamanho do método
Dependências entre métodos
Número de atributos privados/protegidos/públicos
Número de atributos estáticos de uma classe
Número de métodos de uma classe
Dependências entre classes
Resposta para uma classe
Falta de Coesão entre métodos

Tabela 3.3. Métricas segundo Champeaux

Como se percebe, durante a etapa de análise, as métricas se preocupam mais com a avaliação do esforço necessário para o desenvolvimento do sistema, a fim de garantir a viabilidade do projeto. No projeto, a preocupação maior, é com a qualidade do sistema a ser desenvolvido, visando encontrar uma solução que seja adequada para o produto requerido pelo cliente. Já as métricas de implementação, visam tanto observar os aspectos relativos ao esforço gasto para codificação e integração do sistema, quanto observar as características do produto final a ser gerado.

3.6.5 Métricas segundo Abreu

Abreu e Carapuça [1], sugerem uma taxonomia para métricas de produtos e processos OO. Essa taxonomia, chamada de TAPROOT, trabalha em conjunto com o produto e o processo mais algumas métricas híbridas para o tratamento de ambos. A

[8] Descreve os relacionamentos, classes, instâncias de classes e de relacionamentos, além dos subsistemas do projeto.

taxonomia do autor é baseada em um produto cartesiano de dois vetores: um contendo projeto, tamanho, complexidade, reuso, produtividade, qualidade e o outro contendo método, classe e sistema. Esse cruzamento produz 18 possíveis células onde as métricas podem residir. As métricas de qualidade do sistema e das classes que o autor sugere são baseadas na contagem de defeitos observados, falhas e tempo de ocorrência entre as falhas.

3.7 Considerações Finais

Neste capítulo mostrou-se as características desejáveis a uma métrica de software. Em seguida apresentou-se a Teoria de Mensuração [56], que servirá como base para validar a métrica de progresso funcional definida no processo de avaliação de progresso proposto. Esta teoria se fundamenta como o alicerce da métrica, garantindo que ela obedece as propriedades desejáveis a uma métrica, ou justificando o motivo pelo qual alguma das propriedades não foi coberta. Juntamente com a validação teórica da métrica, foi realizado também um estudo de caso sobre um sistema de médio porte (Capítulo 6), de modo a observar a eficácia da métrica sob um ponto de vista mais prático.

Foi apresentada a classificação tradicional das métricas para avaliação do desenvolvimento e da qualidade do produto que está sendo gerado. Em seguida, a classificação específica de métricas para sistemas OO foi considerada. As diversas métricas OO apresentadas, foram classificadas em quatro categorias diferentes, de acordo com Schroeder [51]: tamanho do sistema, tamanho de classe ou método, acoplamento e herança, classes ou métodos internos. Elas apresentam uma visão prática do processo de metrificação utilizado no desenvolvimento de um software OO.

A avaliação de progresso se mostrou uma poderosa técnica de auxílio ao gerenciamento de projeto. Basicamente, se preocupa com o desempenho da equipe de desenvolvimento em relação à execução das atividades, além da inserção de novas funcionalidades, visando a realização dos diversos contratos estabelecidos com o cliente. Apesar da existência de algumas métricas bem definidas, a avaliação de

progresso não consiste em uma atividade simples. A falta de um processo que permita inserir a utilização, a coleta e avaliação das métricas de progresso na organização, impossibilita a utilização eficaz dessas definições, e não garante que o gerente consiga verificar o estado atual do projeto com segurança e confiabilidade.

Uma característica interessante que se pode observar nas visões de métricas dos diversos autores é que, apesar de categorizadas levando-se em conta diferentes aspectos, as métricas consideradas por cada autor se assemelham e apresentam propriedades e significados muitas vezes comuns, como: tamanho, complexidade de uma classe ou método, reuso, produtividade, cronograma, qualidade do produto, herança, coesão e acoplamento. Dentre as diversas categorias apresentadas, se observa que uma grande quantidade de métricas se preocupa ou se relaciona com o progresso no desenvolvimento de um sistema. As métricas definidas neste trabalho visam incorporar as informações providas por várias dessas métricas já existentes, fornecendo uma visão global do progresso do sistema e permitindo uma automação mais direta do processo de avaliação do progresso funcional.

Capítulo 4

Definição da Métrica de Progresso Funcional

Esse capítulo apresenta a definição de uma nova métrica para avaliação de progresso, que visa incorporar as propriedades principais, das diversas métricas de progresso apresentadas na Seção 3.5. A métrica proposta verifica a incorporação de novas funcionalidades dentro do sistema, de acordo com os artefatos que devem ser produzidos para realização de um determinado caso de uso. Dessa forma, o cálculo do progresso é baseado nos casos de uso do sistema em desenvolvimento, que servirão como um guia, ou seja, o progresso do sistema como um todo será obtido a partir do cálculo do progresso funcional de cada caso de uso.

A decisão de se definir uma nova métrica de avaliação de progresso, ao invés de utilizar as existentes, tem o intuito de simplificar a análise dos resultados, sintetizando as informações relacionadas às várias métricas, que indicam o *status* das unidades de trabalho e a capacidade incremental do sistema em um valor único. Além disso, a métrica foi definida de forma a tornar mais simples a avaliação de progresso em processos de desenvolvimento que utilizem o conceito de realização de casos de uso (por exemplo: RUP, OPEN, OOSE).

A métrica $\mu_{sistema}$ representa a métrica global que indica quantitativamente o aspecto funcional já inserido, ou parcialmente inserido, dentro do sistema. Esta métrica servirá como fonte de informação através da qual o gerente de projeto pode fazer diversas considerações como: visualizar o *status* do desenvolvimento das funcionalidades em relação à última avaliação, comparar este *status* com resultados

anteriores, identificar possíveis dificuldades existentes no sistema, identificar atividades mal planejadas, entre outros.

As seções seguintes apresentam a descrição da métrica de progresso funcional definida. Primeiramente são apresentadas as principais características relacionadas com a métrica, observando, de um modo geral, propriedades relacionadas com a forma de se realizar o cálculo da métrica e de se analisar os resultados. É apresentado também o cálculo da métrica, ou seja, o cálculo de $\mu_{sistema}$, que é realizado a partir de equações que capturam o quanto se desenvolveu de um caso de uso (tal captura é feita a partir da inspeção dos artefatos relacionados com a realização de cada etapa necessária para se desenvolver um caso de uso). Em seguida é apresentado um conjunto exemplo de artefatos (retirados a partir de uma análise do RUP) com critérios de inspeção simplificados (tais critérios avaliam os artefatos sintaticamente, sem se preocupar com a corretude) que podem ser aproveitados por organizações que desejem calcular o progresso funcional e possuem um processo de desenvolvimento baseado no RUP. É apresentada também a cobertura desta métrica em relação às diversas métricas de progresso existentes (apresentadas na Seção 3.5). Por fim, são indicadas algumas limitações observadas na métrica de progresso funcional e considerações finais sobre a definição da mesma.

4.1 Características Principais da Métrica de Progresso Funcional

Durante a definição do processo de avaliação de progresso, verificou-se a necessidade de utilizar uma métrica (ou um conjunto de métricas) que permitisse visualizar o estado atual do projeto em termo mais técnico, demonstrando o que já foi desenvolvido e o que ainda falta desenvolver. Visando facilitar o processo de coleta e análise dessa visão de progresso, decidiu-se desenvolver uma métrica global que identifica o progresso funcional do sistema como um todo, permitindo que o gerente realize diversas ponderações, como: comparação com projetos anteriores, cálculo do incremento percentual no progresso do sistema e análise de tendências. A subseções seguintes apresentam as principais características incorporadas à métrica.

4.1.1 Conceito de Progresso Funcional

Como já dito anteriormente, a definição da uma métrica global visa tornar mais prática a avaliação do progresso funcional de um determinado sistema. Inicialmente, a idéia de progresso funcional representa o quanto das funcionalidades previstas para o sistema já foram realizadas [35]. A métrica definida muda esse conceito, fazendo com que o progresso funcional considere, não somente as funcionalidades incorporadas ao sistema, mas, além disso, considere também as funcionalidades que já estão sendo desenvolvidas, mas ainda não foram finalizadas. A métrica observa os artefatos e atividades necessários para o desenvolvimento do sistema, e não apenas a funcionalidade já desenvolvida e testada.

4.1.2 Dirigida a Casos de Uso

Outra característica importante relacionada a essa métrica é o fato dela ser dirigida a casos de uso, ou seja, o cálculo da métrica que indica o progresso funcional do projeto, deriva diretamente do progresso funcional de cada caso de uso do sistema. É feita uma análise do progresso funcional de cada caso de uso onde são considerados os artefatos necessários para desenvolver a funcionalidade representada pelo caso de uso dentro do projeto e, a partir dessas análises individuais é calculado o progresso de todo o sistema.

De forma sucinta, a análise do progresso funcional do caso de uso visa rastrear a realização do caso de uso desde a sua descrição inicial, análise e projeto, até sua implementação e teste, identificando o quanto já se foi feito e o quanto se falta fazer. Assim, durante o rastreamento de um caso de uso são capturados e inspecionados os vários diagramas UML e artefatos relacionados ao seu desenvolvimento.

A Figura 4.1 apresenta um exemplo de diagrama de casos de uso de um projeto X, de forma que a avaliação do progresso deste projeto se dá através da avaliação dos casos de uso que a ele pertencem. Nesse exemplo deverão ser observados e inspecionados os artefatos relacionados com a realização dos casos de uso Cadastrar Tabelas, Atualizar Queixa, Consultar Informações e Especificar Queixa.

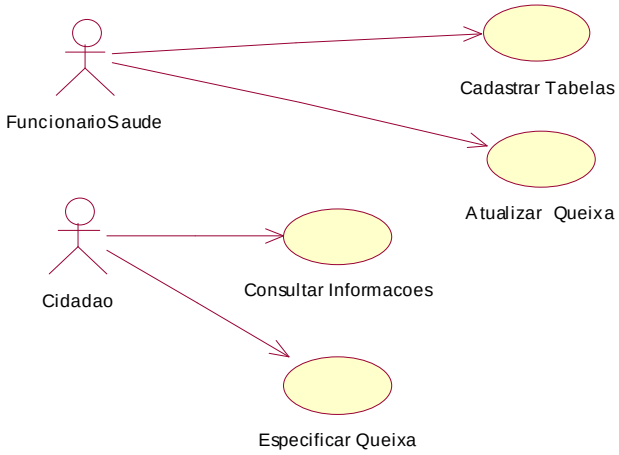


Figura 4.1. Diagrama de casos de uso de um projeto X.

4.1.3 Baseada na Inspeção de Artefatos

Os artefatos correspondem aos documentos envolvidos com a produção do caso de uso e a incorporação das funcionalidades, por ele proposta, dentro do sistema. Exemplos de artefatos relacionados ao caso de uso são: documento de requisitos, diagramas UML, código fonte, casos de teste, entre outros. A métrica considera a produção e/ou atualização de determinados artefatos (os artefatos produzidos pelo processo de desenvolvimento utilizado na organização durante o desenvolvimento de um caso de uso), como sendo a forma de se alcançar o progresso no desenvolvimento de cada caso de uso. É analisado, para cada caso de uso, quais os artefatos foram produzidos ou atualizados para incorporar os objetivos do caso de uso dentro do projeto, e quais ainda terão de ser produzidos ou atualizados. A partir da inspeção destes artefatos, é possível definir o progresso individual de um caso de uso. No trabalho realizado é apresentado uma série de critérios de inspeção sobre os artefatos que não consideram a corretude dos mesmos (Seção 4.3), observando apenas se eles estão sintaticamente corretos. Em uma real aplicação da métrica é necessário um melhoramento de tais critérios, de modo que eles observem os defeitos existentes nos artefatos e não somente verifiquem se os artefatos foram ou não produzidos.

4.1.4 Facilidade de Observação

A métrica definida apresenta como resultado final um valor único, que indica percentualmente o progresso funcional do sistema. Um resultado na forma percentual facilita a comparação com resultados obtidos anteriormente, onde poderão ser feitas comparações diretas dos valores verificando questões de ordem (maior, menor, igual), e a realização de operações aritméticas.

Outra característica relevante que facilita a observação do significado da métrica é a possibilidade de construir gráficos de linha e/ou barra, que permitam realizar comparações e análise de tendências, facilitando ao gerente de projeto a identificação de possíveis problemas no desenvolvimento do projeto e de cada caso de uso, individualmente.

4.2 O Cálculo da Métrica

Como já foi dito, o cálculo da métrica é realizado a partir das contribuições dos diversos casos de uso para o progresso funcional do sistema. A métrica $\mu_{sistema}$, que representa esse tipo de progresso, é derivada da fórmula definida por Champeaux [13], que indica o valor de uma métrica composta, como sendo o somatório da contribuição de cada artefato que compõe a métrica, ou seja:

$$\mu_x = \sum_i D_i \tag{4.1}$$

onde i representa o conjunto de artefatos relevantes à x e D_i representa a contribuição que o artefato i representa para a métrica μ_x. Dessa forma, como o progresso funcional do projeto deriva diretamente do progresso funcional dos diversos casos de uso relacionados ao sistema, temos que a seguinte equação pode ser retirada a partir da Equação 4.1:

$$\mu_{sistema} = \sum_{casodeuso \in DCU} \mu_{casodeuso} \tag{4.2}$$

onde $\mu_{casodeuso}$ indica o progresso de um caso de uso e DCU representa o conjunto do(s) Diagrama(s) de Caso de Uso para o sistema em questão.

O que se observou, inicialmente, na definição dessa métrica, é que ela considerava todos os casos de uso tendo a mesma participação no progresso do sistema. O que se observa na prática é que, casos de uso, frequentemente, apresentam complexidade diferente e necessitam de níveis de esforço variados para sua realização. Dessa forma, atribuíram-se pesos relacionados a cada caso de uso, indicando a prioridade do mesmo para o progresso do sistema. A equação resultante representa uma média ponderada do progresso dos casos de uso, como mostrado a seguir:

$$\mu_{sistema} = \frac{\sum_{casodeuso \in DCU} p_{casodeuso} \times \mu_{casodeuso}}{\sum_{casodeuso \in DCU} p_{casodeuso}} \quad (4.3)$$

onde $p_{casodeuso}$ indica o peso do caso de uso para o progresso do sistema (apresentado com mais detalhes na Seção 4.4), $\mu_{casodeuso}$ indica o progresso do caso de uso, e DCU representa o conjunto do(s) Diagrama(s) de Caso de Uso do sistema.

Outra característica interessante observada durante a definição da métrica, foi a necessidade de se considerar a possibilidade de avaliação de progresso em um processo de desenvolvimento iterativo. O que se deseja aqui é considerar a iteração na qual a avaliação está sendo realizada e, por conseguinte, identificar o progresso em relação ao número de iterações planejadas no plano de projeto. A adaptação da Equação 4.3, nesse momento, definiu a inserção de um parâmetro que indicasse em qual iteração a avaliação de progresso está sendo realizada. Como resultado tem-se:

$$\mu_{sistema}(i) = \frac{\sum_{casodeuso \in DCU} p_{casodeuso}(i) \times \mu_{casodeuso}(i)}{\sum_{casodeuso \in DCU} p_{casodeuso}(i)} \quad com\ 1 \le i \le n \quad (4.4)$$

onde i indica a iteração na qual o projeto se encontra no momento da avaliação, n representa o número de iterações planejadas, e as demais variáveis permanecem com mesmo significado. É importante notar que os valores relacionados ao peso que o caso de uso tem para o sistema podem mudar de iteração para iteração, de acordo com um conjunto de indicadores, que irão demonstrar as prioridades do cliente e os riscos

envolvidos com o desenvolvimento do caso de uso (Seção 4.4). Outra consideração relevante é que, para processos de desenvolvimento que não utilizam o modelo de ciclo de vida iterativo, basta se fazer i igual a 1 para toda avaliação de progresso realizada durante o projeto.

Como $\mu_{sistema}$ deriva diretamente dos progressos dos casos de uso, e estes são calculados a partir da realização das grandes etapas[9] necessárias para seu desenvolvimento, pode-se considerar o progresso de cada uma dessas etapas, ao sistema como um todo. Visando permitir esse tipo de consideração, foi definida a seguinte variação:

$$\mu_{sistema}(i, j) = \frac{\sum_{casodeuso \in DCU} p_{casodeuso}(i, j) \times \mu_{casodeuso}(i, j)}{\sum_{casodeuso \in DCU} p_{casodeuso}(i, j)} \quad (4.5)$$

onde i indica a iteração que o projeto se encontra no momento da avaliação, j representa a etapa de desenvolvimento cujo progresso está sendo avaliado, $p_{casodeuso}(i, j)$ indica o peso relativo ao desenvolvimento da etapa j do caso de uso na iteração i (apresentado com mais detalhes na Seção 4.4), e $\mu_{casodeuso}(i, j)$ indica o progresso da etapa de desenvolvimento j de cada caso de uso, na iteração i.

A partir da definição da métrica global de avaliação de progresso, dirigida a casos de uso, verificou-se a necessidade de se definir $\mu_{casodeuso}(i)$, ou seja, o progresso relativo a um caso de uso. As subseções seguintes visam satisfazer essa necessidade, de modo a identificar o progresso de cada caso de uso.

[9] Um etapa de desenvolvimento de um caso de uso focaliza algum aspecto relacionado a incorporação do caso de uso dentro do sistema. Exemplos de etapas para o desenvolvimento do caso de uso são: especificação, projeto, implementação, teste, entre outros.

4.2.1 Definindo o Progresso de Um Caso de Uso

A métrica relacionada ao progresso de um caso de uso considera as principais etapas necessárias para o seu desenvolvimento, partindo desde a identificação e documentação inicial do caso de uso até seu desenvolvimento e integração. Desse modo temos:

$$\mu_{casodeuso}(i) = \frac{\sum_{j \in ETAPAS} p_{casodeuso}(i, j) \times \mu_{casodeuso}(i, j)}{\sum_{j \in ETAPAS} p_{casodeuso}(i, j)} \quad (4.6)$$

onde i indica a iteração atual, j representa a etapa de desenvolvimento do caso de uso avaliada, $p_{casodeuso}(i, j)$ indica o peso relativo ao caso de uso na iteração i e etapa de desenvolvimento j (apresentado com mais detalhes na Seção 4.4), $\mu_{casodeuso}(i, j)$ indica o progresso do caso de uso na iteração i e etapa de desenvolvimento j, e *ETAPAS* corresponde ao conjunto que contém as etapas necessárias para o desenvolvimento do caso de uso. Um valor padrão definido no trabalho para o conjunto de etapas que devem ser realizadas durante o desenvolvimento de um caso de uso é: *ETAPAS* = {especificação inicial, análise e projeto, implementação, teste}, mas nada impede que esse conjunto possa ser configurado para cada caso de uso de forma a refletir as etapas que realmente necessitam serem realizadas para sua definição, ou seja, podem ser removidas etapas ou inseridas novas etapas de acordo com o processo de desenvolvimento da organização e com a necessidade de se realizar essa etapa de desenvolvimento para um caso de uso especifico.

O progresso de um caso de uso em uma determinada etapa ($\mu_{casodeuso}(i, j)$), é calculado a partir da contribuição dos diversos artefatos necessários para que esta etapa seja realizada. Desse modo, temos a função $\mu_{casodeuso}(i, j) \equiv \mu_{casodeuso}(i, j)(A_{ij})$, que representa a dependência da métrica em relação aos artefatos, com A_{ij} representando o conjunto de artefatos relevantes para realização do caso de uso, na iteração i e etapa de desenvolvimento j.

Novamente, seguindo a Equação 4.1, que consiste na equação de composição proposta por Champeaux [13], definiu-se

$$\mu_{casodeuso}(i, j) = \frac{\sum_{x \in A_{ij}} p_x(casodeuso) \times \mu_x(casodeuso)}{\sum_{x \in A_{ij}} p_x(casodeuso)} \quad (4.7)$$

onde $p_x(casodeuso)$ indica o peso do artefato x para construção de *casodeuso*, $\mu_x(casodeuso)$ representa a métrica que indica percentualmente se o artefato x já foi produzido ou alterado de modo a realizar a etapa j no desenvolvimento do caso de uso, e A_{ij} representa o conjunto de Artefatos que devem ser produzidos ou atualizados na iteração i e etapa j. A configuração desse conjunto de artefatos necessários em cada etapa de um caso de uso deverá ser realizada durante o planejamento da avaliação de progresso. Cada artefato identificado deve conter critérios de avaliação, que permitam identificar se tais artefatos incorporam as informações necessárias ao desenvolvimento do caso de uso (seja completamente ou parcialmente).

A partir de um estudo realizado sobre o RUP (*Rational Unified Process*) [7, 48], que consiste em um processo de desenvolvimento OO, iterativo e incremental, definido pela Rational, se definiu um padrão inicial de artefatos, com seus respectivos critérios de avaliação, adequado para esse processo de desenvolvimento. Como foi dito, este conjunto é apenas um padrão apresentado para uma possível aplicação da métrica em organizações que utilizem o RUP como processo de desenvolvimento, caso a organização não utilize o RUP, esse conjunto pode ser reconfigurado de forma a se tornar mais adequado ao processo de desenvolvimento que ela utiliza.

A escolha inicial do RUP se deve ao fato do mesmo ser um processo que utiliza a linguagem de modelagem UML (com conceitos como realização dos casos de uso) e por ele ser um processo de desenvolvimento guiado através da definição dos casos de uso. Como resultado dessa análise do RUP foi definida a Tabela 4.1, contendo os principais artefatos produzidos durante cada etapa do desenvolvimento de um caso de uso.

Etapa no Desenvolvimento	Artefato	Breve Descrição
Especificação Inicial	Diagrama de Caso de Uso	Diagrama UML que contém os casos de uso do sistema e seus relacionamentos com outros casos de uso e atores.
	Descrição Funcional	Representa uma descrição do que se espera que o caso de uso faça, o que ele espera receber como entrada e produzir como saída.
	Fluxo de Eventos	Representa um conjunto de cenários que demonstram o comportamento do caso de uso, de acordo com eventos externos.
	Artefatos de Protótipo	Artefatos relacionados à definição, projeto e implementação de um protótipo de interface do sistema.
Análise e Projeto	Diagrama de Sequência/Colaboração	Mostram interações entre objetos e atores dentro do sistema, incluindo as mensagens que são disparadas entre eles.
	Diagrama de Classe	Define um conjunto de classes e interfaces, indicando colaborações e relacionamentos. Representa a visão estática do projeto do sistema.
	Diagrama de Estado	Máquina de estado, consistindo de estados, transições, eventos e atividades. Modela o comportamento de um objeto.
	Diagrama de Atividade	Tipo especial de diagrama de estado que mostra um fluxo de atividades dentro do sistema. Importante para modelagem de funções do sistema.
Implementação	Diagrama de Componente	Mostra a organização e as dependências entre um conjunto de componentes. Endereça a visão estática da implementação do sistema.
	Código Fonte	Código do sistema contendo a implementação de classes e métodos.
	Resultado dos Testes de Unidade	Resultado dos testes visando identificar problemas em uma determinada classe.
	Versão Executável do Sistema	Versão contendo as últimas modificações realizadas no sistema.
Teste	Plano de Testes	Documento contendo estratégias para teste do sistema como um todo.
	Casos de Teste	Testes derivados diretamente do caso de uso, visando identificar problemas na sua realização.
	Procedimentos de Teste	Procedimentos necessários para implementação dos testes.
	Scripts de Teste	Programas auxiliares desenvolvidos para a realização do teste.

Tabela 4.1. Artefatos padrões envolvidos com o desenvolvimento de um caso de uso

Cada artefato identificado na Tabela 4.1 deve conter um conjunto de critérios de avaliação agrupados, que servirão como fonte de consulta durante a inspeção do artefato visando identificar se o mesmo incorpora as funcionalidades do caso de uso. Uma dificuldade inerente a esse processo de inspeção está na avaliação semântica dos artefatos, ou seja, na avaliação da corretude dos mesmos, que não foi coberta pelo

trabalho devido à complexidade desse tipo de análise e por não ser o foco do trabalho definir tais critérios de inspeção. Desse modo, não foram focalizadas questões semânticas como defeitos nos artefatos, verificando-se apenas as características sintáticas que o artefato deve apresentar e o relacionamento deste com os demais artefatos. Existem trabalhos relacionados que se preocupam em observar os artefatos em busca de defeitos no desenvolvimento e que podem ser considerados durante a definição dos critérios de inspeção de artefatos para uma organização específica [54].

É interessante observar que na análise realizada, os artefatos do fluxo de implantação do RUP não foram considerados como sendo parte do desenvolvimento de um caso de uso. Isso aconteceu pelo fato da implantação não ser realizada sobre cada caso de uso individualmente, mas sim sobre todo o sistema. Desse modo, para que o progresso na implantação do sistema seja capturado, é importante definir outras métricas que observam a implantação do sistema como um todo.

A Seção 4.3, apresenta os critérios de avaliação, que deverão ser observados, durante a inspeção de cada artefato citado na Tabela 4.1.

4.3 Inspeção dos Artefatos

As subseções seguintes definem um exemplo de conjunto de diretrizes para inspeção parcial dos artefatos padrões definidos para avaliação da realização de um caso de uso. Tais critérios realizam somente uma avaliação sintática dos casos de uso, sem se preocupar com a detecção de defeitos nos artefatos localizados. Para que esse exemplo de conjunto de diretrizes seja realmente eficaz, deverão ser inseridos mecanismos de inspeção que visam tornar mais completa a avaliação dos artefatos. Trabalhos relacionados que se preocupam em observar os artefatos em busca de defeitos no desenvolvimento podem ser encontrados em [54].

Para simplificar o exemplo de conjunto de critérios de inspeção, decidiu-se simplificar os resultados da inspeção de um artefato para 0 ou 1 (artefato não produzido ou artefato produzido, respectivamente), sem considerar valores intermediários que em um ambiente de inspeção mais complexo poderiam existir.

Os artefatos estão subdivididos, entre as subseções, de acordo com a etapa com a qual ele se relaciona durante o desenvolvimento do caso de uso. Os artefatos apresentados nessa seção, são discutidos com mais detalhes na versão estendida do Inspector [37]. Além disso, é importante salientar que esse conjunto de artefatos não é fixo, podendo ser configurado de acordo com o processo de desenvolvimento da organização, onde novos critérios de inspeção sobre os artefatos apresentados podem ser inseridos e ainda novos artefatos com critérios associados podem ser incluídos.

4.3.1 Especificação Inicial

Durante a análise do progresso da etapa de especificação inicial, foram identificados quatro possíveis artefatos relacionados à sua realização. Nesse momento, a preocupação é identificar o caso de uso e certificar-se que o mesmo apresenta uma descrição inicial, contendo a descrição das funcionalidades e eventos relacionados ao caso de uso. Os itens abaixo descrevem um exemplo simples de como verificar se esses artefatos realmente representam a descrição inicial do caso de uso (sem considerar questões semânticas como corretude):

Diagrama de Caso de Uso

Basicamente, deve-se verificar se o caso de uso está inserido no diagrama de caso de uso, com as relações com outros casos de uso e atores definidas. Em caso positivo temos que $\mu_{DCU}(uc) = 1$, caso contrário $\mu_{DCU}(uc) = 0$, onde DCU indica o Diagrama de Caso de Uso e *uc* o caso de uso que está sendo avaliado.

Descrição Funcional

Visa identificar se está definida, no documento de requisitos, uma seção específica para o caso de uso, contendo os principais itens da descrição funcional do caso de uso. Dentre estes itens temos: a descrição detalhada do caso de uso, as entradas permitidas pelo caso de uso, as saídas esperadas, pré-requisitos e pós-requisitos relativos à execução do caso de uso. Para cada item citado, atribuí-se o valor 1 caso o mesmo já tenha sido documentado, caso contrário, atribuí-se o valor 0. O resultado da inspeção da descrição funcional se dá através da equação:

$$\mu_{DescriçãoFuncional}(uc) = \frac{\sum_{x \in \ Itens\ da\ Descrição} inspeção(x)(uc)}{\#\ Itens\ da\ Descrição} \quad (4.8)$$

onde $inspeção(x)(uc)$ é a função que retorna se o item x foi ou não desenvolvido para realização do caso de uso *uc*, e *Itens da Descrição* é o conjunto de itens que devem estar contidos na descrição funcional do caso de uso.

Fluxo de Eventos

Deverá ser verificado se o fluxo de eventos do caso uso está documentado, seja na forma textual ou na forma gráfica (através de diagramas de atividades). É importante observar se o fluxo principal foi definido, se fluxos alternativos e subfluxos são necessários, e se eles estão, ou não, documentados. Atribui-se o valor 1 para os itens desenvolvidos e 0 para os que ainda precisam ser feitos. O resultado da avaliação do fluxo de eventos se dá através da equação:

$$\mu_{FluxodeEventos}(uc) = \frac{\sum_{x \in \ Itens\ do\ FluxodeEventos} inspeção(x)(uc)}{\#\ Itens\ do\ FluxodeEventos} \quad (4.9)$$

onde $inspeção(x)(uc)$ é a função que retorna se x foi ou não desenvolvido para realização do caso de uso *uc*, e *Itens do FluxodeEventos* é o conjunto de itens (fluxo principal, fluxo alternativo, subfluxos) que devem ser inspecionados durante a avaliação dos fluxos relacionados ao caso de uso.

Protótipo de Interface

Se houver necessidade de incorporar as funcionalidades do caso de uso em um protótipo inicial, temos que deverá ser verificado se cenários de uso foram definidos, se o projeto de interface do caso de uso foi realizado e se o protótipo contém as interfaces que representarão a funcionalidade do caso de uso. Para cada resposta positiva, temos que deverá ser atribuído o valor 1, em caso negativo será atribuído o valor 0. O resultado da avaliação do protótipo de interface se dá através da equação:

$$\mu_{ProtótipodeInterface}(uc) = \frac{\sum_{x \in \ Itens\ do\ ProtótipodeInterface} inspeção(x)(uc)}{\#\ Itens\ do\ ProtótipodeInterface} \quad (4.10)$$

onde $inspeção(x)(uc)$ é a função que retorna se x foi ou não desenvolvido para realização do caso de uso *uc*, e *Itens do ProtótipodeInterface* é o conjunto de itens que devem ser desenvolvidos durante a definição e implementação de um protótipo de interface.

4.3.2 Análise e Projeto

Durante a análise do progresso da etapa de análise e projeto do caso de uso, é necessário avaliar os diversos diagramas UML relacionados à realização da modelagem do caso de uso: diagramas de interação (sequência/colaboração), diagrama de classe, diagrama de atividade (se necessário), diagrama de estado (se necessário). O diagrama de componentes não é considerado nessa etapa, pois se relaciona com a implementação do caso de uso.

Diagrama de Interação

Este tipo de diagrama tem como objetivo mostrar o comportamento dos vários objetos relacionados, durante a realização do caso de uso. Ele é classificado em dois tipos de diagramas, de acordo com o foco que ele demonstra: diagrama de Sequência (foco na sequência das mensagens trocadas entre os objetos) e diagrama de Colaboração (foco no comportamento de cada objeto dentro do caso de uso) [8]. Ambos os diagramas podem representar os aspectos dinâmicos relativos à modelagem do caso de uso, indicando as interações dos usuários com o sistema, e a troca de mensagens entre as diversas classes relacionadas ao caso de uso para produzir o resultado desejado. A inspeção de cada um dos diagramas de sequência e/ou colaboração relacionados ao caso de uso obedece aos seguintes critérios:

- Deve haver um rastreamento direto entre o caso de uso e o diagrama de interação em observação, ou seja, o diagrama de interação deve pertencer à realização do caso de uso, ou estar imediatamente abaixo do caso de uso na hierarquia do

projeto (caso se esteja utilizando alguma ferramenta que permita esse tipo de ligação, ex.: Rational Rose).

- O diagrama deve ter o nome de identificação do caso de uso associado ao seu nome.
- Deve representar as diversas classes de análise envolvidas com a realização do caso de uso e que modelam o comportamento do mesmo.
- Deve representar as classes de controle que monitoram as classes de análise, de forma a obter os resultados desejados.
- Deve conter mensagens que permitem realizar os fluxos de eventos descritos para o caso de uso.

Caso todos esses critérios sejam avaliados positivamente, temos que $\mu_{diagrama\ de\ iteração}(uc) = 1$, caso contrário $\mu_{diagrama\ de\ iteração}(uc) = 0$, onde *uc* é o caso de uso em avaliação.

Diagrama de Classe

Esse diagrama representa a estrutura das diversas classes relacionadas à realização do caso de uso, indicando também os relacionamentos entre tais classes [8]. A inspeção desse diagrama, visando identificar as diversas classes necessárias para realização do caso de uso, obedece os seguintes critérios:

- As classes de análise definidas no diagrama de interação devem estar representadas no diagrama de classes contendo: atributos (possíveis parâmetros passados nas mensagens do diagrama de interação, ou ainda atributos relativos a análise do caso de uso), métodos (correspondem às mensagens trocadas nos diagramas de interação e métodos de acesso aos atributos do sistema) e relacionamentos com outras classes (resultantes do comportamento da classe para atingir a funcionalidade do caso de uso).
- As classes de controle definidas no diagrama de sequência do caso de uso, devem estar representadas no diagrama de classes, contendo os atributos e métodos (mensagens que ela envia) necessários, para controlar a execução do caso de uso.

- As classes de fronteira relacionadas com a prototipação do caso de uso, devem estar representadas no diagrama de classe, contendo atributos, métodos relevantes e relacionamentos com outras classes bem definidos.

Uma observação importante relacionada com a inspeção desse diagrama é que a localização de uma classe em um pacote ou subsistema não afeta o progresso funcional do sistema, pois ela continua com as mesmas propriedades, e realizando as mesmas operações. Tem-se também que as classes podem ser alteradas de modo a adquirir novas funcionalidades relevantes a outros casos de uso. Caso todos os critérios acima definidos sejam avaliados positivamente, temos que $\mu_{diagrama\ de\ classe}(uc) = 1$, caso contrário $\mu_{diagrama\ de\ classe}(uc) = 0$, onde *uc* é o caso de uso em avaliação. Para simplificar o exemplo de conjunto de critérios de inspeção, decidiu-se simplificar os resultados para 0 ou 1, sem considerar valores intermediários, que em um ambiente de inspeção mais complexo poderiam existir.

Diagrama de Estado

Consiste em uma máquina de estado, contendo estados, transições, eventos e atividades [8]. Endereça a visão dinâmica do sistema. Importante para modelar um objeto com comportamento dinâmico, enfatizando os eventos que resultam em mudança de estado para o objeto.

Quando necessário, ou seja, quando o caso de uso possuir algum objeto relacionado, que contém a necessidade de uma modelagem dos seus diversos estados durante a realização do caso de uso, temos que os seguintes critérios devem ser obedecidos:

- Caso exista a ferramenta de modelagem que permita manter o relacionamento do caso de uso com sua realização, basta identificar, para os objetos que necessitam, se existe o diagrama de estado correspondente.
- Deve existir um diagrama de estado, para cada objeto relativo ao caso de uso, que necessite de uma modelagem dos seus estados, ou seja, apresente um comportamento bastante dinâmico, derivado da realização do caso de uso.

- Alguns desses estados devem representar o comportamento do objeto no caso de uso, ou seja, estados oriundos da realização do caso de uso.

Caso todos esses critérios sejam avaliados positivamente, temos que $\mu_{diagrama\ de\ estado}(uc) = 1$, caso contrário $\mu_{diagrama\ de\ estado}(uc) = 0$, onde *uc* é o caso de uso em avaliação.

Diagrama de Atividade

Tipo especial de diagrama de estado que mostra um fluxo de atividades dentro do sistema. Endereça a visão dinâmica do sistema. Importante para modelagem de uma função de um sistema, focalizando o fluxo de controle entre objetos. Quando necessário, ou seja, quando o caso de uso possuir um objeto com alguma operação complexa que necessite ser modelada, temos que os seguintes critérios devem ser observados:

- Caso exista a ferramenta de modelagem que permita manter o relacionamento do caso de uso com sua realização deve-se rastrear o diagrama de atividade correspondente à realização de alguma operação complexa dentro do caso de uso.
- Quando houver necessidade, a sequência de atividades para realização de uma determinada operação, envolvida com o caso de uso, deve estar modelada em um diagrama de atividade.

Caso esses critérios sejam avaliados positivamente, temos que $\mu_{diagrama\ de\ atividade}(uc) = 1$, caso contrário $\mu_{diagrama\ de\ atividade}(uc) = 0$, onde *uc* é o caso de uso em avaliação.

4.3.3 Implementação

Verifica se o diagrama de componentes do sistema, contém as classes derivadas do caso de uso, observa se as classes definidas foram codificadas, se testes de unidade sobre essas classes já foram realizadas e se a versão executável do sistema (se houver) disponibiliza para uso, as funcionalidades previstas no caso de uso. Abaixo seguem os

critérios de avaliação da realização dos artefatos relacionados, para um determinado caso de uso.

Diagrama de Componente

- Deve haver um mapeamento direto das classes derivadas do caso de uso com os componentes existentes no diagrama de componentes
- Bibliotecas de interface, classes de fronteira e comunicação devem estar representadas

Caso todos esses critérios sejam avaliados positivamente, temos que $\mu_{diagrama\ de\ componente}(uc) = 1$, caso contrário $\mu_{diagrama\ de\ componente}(uc) = 0$, onde *uc* é o caso de uso em avaliação.

Código Fonte

- Visa identificar se as classes derivadas do caso de uso foram implementadas com as funcionalidades (atributos e métodos) definidas durante a realização da análise e projeto do caso de uso

Caso o critério acima seja avaliado positivamente, temos que $\mu_{codigo\ fonte}(uc) = 1$, caso contrário $\mu_{codigo\ fonte}(uc) = 0$, onde *uc* é o caso de uso em avaliação.

Resultado dos Testes de Unidade

- Representa a porcentagem de classes derivadas do caso de uso que já realizaram seus testes de unidade, ou seja

$$\mu_{teste\ de\ unidade}(uc) = \frac{classes\ \ derivadas\ do\ caso\ de\ uso\ \ já\ testadas}{classes\ \ derivadas\ do\ caso\ de\ uso} \qquad (4.11)$$

onde *uc* é o caso de uso em avaliação. Os critérios que podem ser utilizados para avaliar se um teste de unidade foi realizado sobre um caso de uso são:

- O teste é identificado
- Casos de teste são definidos (derivados a partir do caso de uso, para a classe)

- Procedimentos de teste são descritos
- Testes de unidades implementados, executados e positivamente avaliados

Interface

- Verifica se as funcionalidades definidas pelo caso de uso estão disponíveis para uso na interface do sistema

Caso o critério acima seja avaliado positivamente, temos que $\mu_{\text{int}\,erface}(uc) = 1$, caso contrário $\mu_{\text{int}\,erface}(uc) = 0$, onde *uc* é o caso de uso em avaliação.

4.3.4 Teste

Consiste em inspecionar se o caso de uso está integrado ao sistema, observando se os testes de sistema e de integração foram planejados, projetados, *scripts* de teste foram implementados e executados, verificando também a porcentagem de testes que foram realizados com sucesso (sem considerar questões semânticas como corretude dos testes realizados).

Plano de Teste

- Consiste em verificar se os casos de teste para o caso de uso foram identificados e agrupados;
- Se os riscos inerentes foram levantados;
- Se estratégias de teste foram definidas;
- Se os critérios de avaliação foram estabelecidos; e
- Se os recursos de teste foram identificados e o cronograma de teste já foi detalhado e documentado.

Caso todos esses critérios sejam avaliados positivamente, temos que $\mu_{plano\ de\ teste}(uc) = 1$, caso contrário $\mu_{plano\ de\ teste}(uc) = 0$, onde *uc* é o caso de uso em avaliação.

Projeto de Teste

- Verifica se os casos de teste identificados foram descritos;

- Se os procedimentos de teste foram estruturados; e
- Se métricas de avaliação dos testes foram descritas.

Caso todos esses critérios sejam avaliados positivamente, temos que $\mu_{projeto\ de\ teste}(uc) = 1$, caso contrário $\mu_{projeto\ de\ teste}(uc) = 0$, onde *uc* é o caso de uso em avaliação.

Implementação do Teste

- Verifica se os scripts de teste dos casos de teste anteriormente identificados foram gravados ou programados;
- Se o conjunto de dados externos (necessários para os testes) foram criados e mantidos; e
- Se os pacotes e classes de teste relativos aos casos de teste envolvidos foram projetados e implementados.

Caso todos esses critérios sejam avaliados positivamente, temos que $\mu_{scripts\ de\ teste}(uc) = 1$, caso contrário $\mu_{scripts\ de\ teste}(uc) = 0$, onde *uc* é o caso de uso em avaliação.

Execução do Teste

- Indica a porcentagem dos casos de teste sobre o caso de uso que foram realizados com sucesso, ou seja, a porcentagem dos casos de teste cujas métricas de avaliação definidas no plano de teste indicaram uma realização satisfatória. Desse modo, temos que:

$$\mu_{execucao\ dos\ testes}(uc) = \frac{testes\ executados\ com\ sucesso}{testes\ executados} \qquad (4.12)$$

onde *uc* é o caso de uso em avaliação.

4.4 Indicadores para Definição dos Pesos

O cálculo da métrica global, que indica o progresso do sistema, envolve a definição de um conjunto de pesos, relacionados ao desenvolvimento do caso de uso, às etapas

necessárias para o seu desenvolvimento, e aos artefatos que devem ser desenvolvidos em cada etapa. A identificação desses pesos não consiste em um trabalho simples. É necessário definir critérios para sua definição, ou basear-se em experiências com projetos anteriores, que apresentem características similares. Essa segunda opção, muitas vezes, não é possível ser utilizada, pois a organização talvez não possua experiências anteriores com projetos de porte semelhante, que tenham usado tecnologia similar, pessoal de mesmo nível técnico, entre outros fatores. Desse modo, achou-se interessante definir, resumidamente, alguns indicadores simples, que servirão como fonte de consulta para definição dos pesos associados ao projeto. Esses indicadores não necessitam ser obrigatoriamente utilizados, podendo novos indicadores serem definidos, como forma de se encontrar os diversos pesos associados à $\mu_{sistema}$.

4.4.1 Pesos dos Casos de Uso

Na métrica $\mu_{sistema}$, temos que, para cada caso de uso, existe um peso associado ($p_{casodeuso}$), que é influenciado pela complexidade da realização do caso de uso, ou seja, o quanto de esforço vai ser exigido na construção do caso de uso. Alguns fatores que influenciam na definição da prioridade do caso de uso no sistema, são:

- Qual a prioridade do caso de uso para o cliente? Casos de uso com maior prioridade para o cliente, possivelmente também terão maior prioridade durante o desenvolvimento.

- Quais os riscos envolvidos com a implementação do caso de uso? Casos de uso que apresentam grande quantidade de riscos envolvidos (especialmente, riscos técnicos, ou seja, riscos relacionados com a tecnologia utilizada), possivelmente irão possuir maior prioridade durante o desenvolvimento.

- Pode-se fazer uma estimativa de esforço e tempo sobre o caso de uso, como por exemplo, análise por pontos de função [10]. De acordo com o resultado, o peso associado à métrica é atribuído.

4.4.2 Pesos das Etapas dos Casos de Uso

Nesse caso, os pesos associados, se referem às dificuldades inerentes e ao esforço necessário para o desenvolvimento de cada etapa do caso de uso. As etapas padrões, apresentadas para realização de um caso de uso, são: especificação inicial, análise e projeto, implementação e teste. Alguns indicadores para a definição dos pesos associados a cada etapa são:

- Identificar o esforço médio (tempo de desenvolvimento em relação ao número de pessoas envolvidas) associado à realização de cada etapa de desenvolvimento (análise, projeto, implementação, entre outras) de um caso de uso. Esse esforço médio pode ser identificado a partir de valores obtidos em projetos anteriores realizados na organização. Etapas que exigem maior esforço devem possuir um maior peso associado.

- Utilizar valores padrões, disponíveis em artigos e trabalhos na área, onde foram realizadas diversas experiências, visando buscar a distribuição do esforço e do tempo para o desenvolvimento de um sistema. Por exemplo, Capers Jones em [31], apresenta uma distribuição do esforço na etapa de implementação, de acordo com a linguagem de programação utilizada. Outro exemplo, é a distribuição típica do esforço associado a cada fluxo de desenvolvimento, durante a realização das diversas iterações, no RUP [7]. Esse indicador deve ser utilizado com cuidado, pois os valores padrões dos trabalhos podem estar relacionados com um tipo de aplicação ou a uma organização específica. É necessário, desse modo, observar se o projeto no qual a métrica vai ser aplicada é adequado para aplicar tais valores.

- O nível de conhecimento e experiência de uma determinada equipe, com a tecnologia necessária para se realizar uma determinada etapa, também é um fator importante. Por exemplo, o projeto de um caso de uso exigirá um alto nível de esforço para uma equipe com pouco conhecimento em UML (*Unified Modeling Language*). Nesse caso, o progresso em se fazer o projeto do caso de uso irá ser maior do que realizar uma especificação inicial do mesmo.

- Identificar, no momento da coleta, quais etapas do caso de uso têm apresentado (ou apresentaram) as maiores dificuldades para seu desenvolvimento, e prever as maiores dificuldades que ele ainda vai enfrentar, através de uma análise de sua especificação inicial.

4.4.3 Pesos dos Artefatos Relacionados

Para realizar cada etapa do caso de uso, temos que vários artefatos devem ser atualizados e/ou produzidos. Cada artefato desse possui um esforço associado ao seu desenvolvimento e, às vezes, a documentação de um artefato mais complexo representa um progresso maior do que a documentação de um artefato simples e rápido. Desse modo, é interessante definir o peso do artefato, a partir da complexidade de se desenvolver o mesmo. Alguns indicadores para se observar a complexidade do artefato são:

- Identificar o tempo médio gasto para produção e/ou atualização desse artefato em projetos anteriores. De acordo com esse tempo, é possível identificar o esforço gasto para sua produção. A documentação de um artefato que precisa de mais tempo para ser produzido do que um outro artefato, possivelmente representa um progresso maior para realização do caso de uso do que a documentação do artefato que necessita de menos tempo.

- Verificar o tamanho do artefato associado. A produção de artefatos extensos pode indicar um progresso maior do que a produção de um documento menor pois, após a produção destes artefatos maiores, o tempo para se finalizar o projeto poderia cair substancialmente. Vale salientar que esse indicador deve ser usado com critério já que, muitas vezes, um diagrama UML (ocupando um espaço pequeno) possui uma complexidade muito maior do que a elaboração da descrição inicial do caso de uso (que possui tamanho maior).

- O número de pessoas associadas com a produção de um artefato também pode ser um indicador da complexidade do mesmo. Artefatos com grande número de

pessoas, possivelmente são mais complicados, trabalhosos e leva mais tempo para serem desenvolvidos, devendo possuir um peso maior atribuído ao mesmo.

4.5 Formas de Representação de $\mu_{sistema}$

A métrica apresenta uma grande variedade de considerações que podem ser trabalhadas para avaliação do progresso funcional do projeto como um todo, e também de cada caso de uso individualmente. Uma métrica que não permite ser avaliada de maneira clara, nem permite a comparação de valores atuais com valores anteriores e/ou estimados, não representa uma grande contribuição para o gerente de projeto. É quase um consenso que a forma de representação gráfica consiste na melhor maneira de se representar uma métrica, pois permite uma melhor observação da variação da mesma durante o tempo [5].

A métrica de avaliação do progresso funcional apresenta duas formas de representação: tabela e gráficos. A tabela resume todos os dados relativos ao estado atual do projeto e dos seus diversos casos de uso considerando as etapas para realização do mesmo. Já a forma de representação gráfica considera o progresso de apenas um elemento do projeto ou do projeto como um todo. Desse modo, deve haver um gráfico de linha associado a cada caso de uso e ao projeto como um todo indicando seu progresso em relação ao tempo.

4.5.1 Representação em Forma de Tabelas

Como já mostrado, a métrica global do progresso funcional do projeto deriva diretamente do progresso de cada caso de uso que, por sua vez, é derivado do progresso na realização de cada etapa do caso de uso. Uma forma de representação dos resultados obtidos durante o cálculo da métrica é uma tabela única que sintetiza o progresso de cada caso de uso e o progresso do projeto.

O número de linhas da tabela é definido a partir do número de casos de uso do sistema onde, para cada caso de uso, haverá uma linha representando seu progresso. A última linha indica o progresso do projeto como um todo. Já as colunas são definidas a

partir do número de etapas necessárias para realização do caso de uso, ou seja, do conjunto ETAPAS, que é configurado durante a instanciação do processo de avaliação de progresso. A Tabela 4.2 representa um exemplo de como os dados são sintetizados para mostrar uma visão global do projeto em um instante t.

Caso de Uso	Especificação Inicial	Análise e Projeto	Implementação	Teste	Progresso Total
1	0,66	0,66	0	0	**0,33**
2	1	1	0,66	0,50	**0,79**
3	0,75	0,66	0	0	**0,35**
4	0,75	0,66	0	0	**0,35**
5	1	1	1	1	**1**
Projeto	**0,94**	**0,92**	**0,64**	**0,58**	**0,77**

Tabela 4.2. Exemplo de tabela gerada a cada cálculo da métrica

Pode ser observado na Tabela 4.2 o progresso funcional de todos os casos de uso, identificando além do progresso total, o progresso encontrado na inspeção dos artefatos relacionados a cada etapa de desenvolvimento do caso de uso. Por exemplo, a tabela indica que para o caso de uso 1 o progresso na etapa de especificação inicial é de 0,66 (66% dos artefatos relacionados a essa etapa foram produzidos e passaram pelos critérios de inspeção utilizados). Ainda para o caso de uso 1 observa-se que no momento da avaliação não houve a produção de nenhum artefato relacionado ao seu teste (0%).

Este tipo de representação é adequado para identificar problemas em casos de uso específicos, problemas na realização de alguma etapa de desenvolvimento do caso de uso, e ainda, manter um histórico preciso do projeto em desenvolvimento, servindo como base de conhecimento para projetos futuros e para as próximas avaliações de progresso.

4.5.2 Representação em Forma de Gráficos de Linha

Gráficos de linha consistem em outra forma de representação dos resultados calculados. A escolha deste tipo de gráfico veio pelo fato do tempo ser uma variável importante na avaliação do progresso técnico de um projeto, sendo possível representar em um único gráfico de linha os resultados obtidos nas avaliações de progresso realizadas em momentos diferentes. Tais gráficos consideram o progresso de cada caso

de uso individualmente, ou seja, é necessário um gráfico para representar o progresso de cada caso de uso.

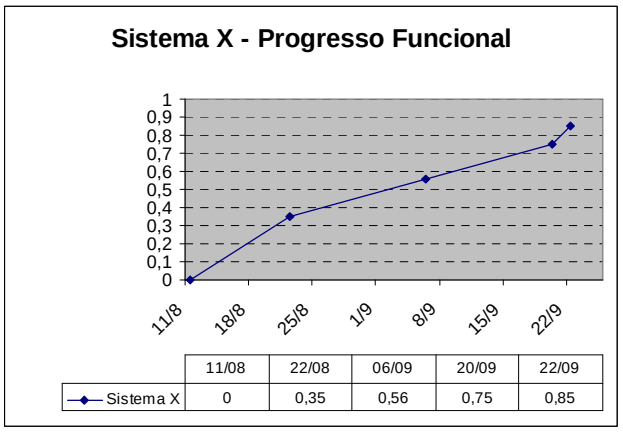
Como já dito, a principal característica inserida nessa forma de representação é que o tempo é considerado, ou seja, o gráfico recupera toda a evolução do caso de uso, desde o início do projeto. Desse modo, os valores da métrica que foram calculados em datas de avaliação anteriores são mantidos e recuperados para geração do gráfico, indicando o progresso do caso de uso durante o transcorrer do projeto. Além de gráficos para cada caso de uso, é gerado também o gráfico relativo ao progresso global do projeto em relação ao tempo. A Figura 4.2 representa exemplos de gráficos de linha gerados para um sistema X, o gráfico A representa o resultado obtido a partir de um conjunto de avaliações de progresso sobre o projeto como um todo, já o gráfico B indica o progresso de um caso de uso específico deste projeto.

Os gráficos de linha também são bastante úteis como fonte de informação para futuros projetos, onde é possível identificar a duração prevista para o projeto, a partir da duração de projetos anteriores com porte semelhante, e observar os problemas enfrentados, para que não se caia nos mesmos erros.

4.6 Validação da Métrica segundo a Teoria de Mensuração

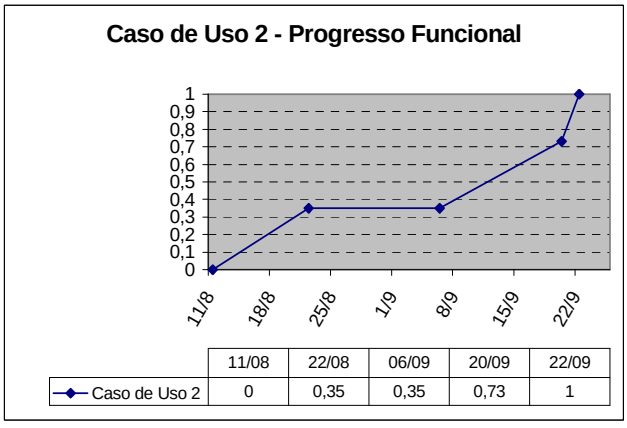
Nos últimos tempos, muitas métricas vêm sendo definidas e utilizadas, mas não existe nenhuma comprovação teórica que tais métricas representam algum tipo de contribuição, e se elas realmente quantificam o que inicialmente motivou a sua definição. Desse modo, surgiu a necessidade de se realizar uma comprovação teórica da métrica proposta. As subseções seguintes identificam quais propriedades $\mu_{sistema}$ possui, e quais não são cobertas, segundo a Teoria de Mensuração apresentada em Zuse [56].

Gráfico A



	11/08	22/08	06/09	20/09	22/09
Sistema X	0	0,35	0,56	0,75	0,85

Gráfico B



	11/08	22/08	06/09	20/09	22/09
Caso de Uso 2	0	0,35	0,35	0,73	1

Figura 4.2. Exemplos de gráfico de linha gerados a partir de avaliações de progresso

4.6.1 Sistemas Relacionais

Seja **Projeto** = (*Projetos*, R_i), o par formado pelo conjunto de projetos *Projetos*, e o conjunto de relações empíricas R_i permitidas entre esses projetos (maior, menor, igual). Além disso, seja a representação formal **Progresso do Projeto** = (*Progresso do Projeto*, $= | \leq | \geq | < | >$), onde *Progresso do Projeto* representa um conjunto não vazio de números, com cada número desse indicando o progresso de um elemento de *Projetos*. Nesse caso, a métrica $\mu_{sistema}$, representa uma função de mapeamento, onde $\mu_{sistema} : Projetos \rightarrow Progresso \quad do \quad Projeto$, que coloca o valor de $\mu_{sistema}(projeto)$ em *Progresso do Projeto* para todo projeto do conjunto *Projetos*. Temos então que:

$$\forall p \in Projetos \; \exists \; g \in Progresso \; do \; Projeto \;\middle|\; \mu_{sistema}(p) = g$$

Além disso, as relações empíricas R_i = {maior, menor, igual}são mapeadas para as relações matemáticas $= \left| \leq \right| \geq \left| < \right| >$. Logo, pode-se concluir que $\mu_{sistema}$ representa um sistema relacional que mapea uma relação empírica em um sistema relacional formal.

4.6.2 Ordem

Essa propriedade indica que o sistema relacional formal, definido pela métrica, deve apresentar seus elementos através de uma ordem parcial, ou seja, apresentar as propriedades reflexiva ($aRa, \forall a \in A)$, transitiva $(aRb \wedge bRc \rightarrow aRc, \forall a,b,c \in A)$ e anti-simétrica $(aRb \; e \; bRa \rightarrow a = b, \forall a,b \in A)$. Como já mostrado na seção anterior, $\mu_{sistema}$ resulta no sistema relacional formal **Progresso do Projeto** = (*Progresso do Projeto,* $= \left| \leq \right| \geq \left| < \right| >$), onde $\forall g \in Progresso \; do \; Projeto$ temos $g = \mu_{sistema}(p)$, com $p \in Projetos$ e g consiste em um valor numérico entre 0 e 1 que indica o progresso funcional do projeto.

Podemos observar que o sistema relacional definido pela métrica possui a propriedade reflexiva, pois contém a relação "=", de tal forma que, $\forall g \in Progresso \; do \; Projeto$, temos que $g = g$. A propriedade transitiva também é obedecida, pois, dados $g1, g2, g3 \in Progresso \; do \; Projeto$, temos que existem as relações "=", ">", "<", "≥", "≤" que garantem essa propriedade sobre valores numéricos. Por exemplo, se $g1 > g2$ e $g2 > g3$, temos que $g1 > g3$. Por fim, a propriedade anti-simétrica também é coberta, pois dados $g1, g2 \in Progresso \; do \; Projeto$, temos que existem as relações "≥", "≤", que garantem essa propriedade sobre valores numéricos. Por exemplo, se $g1 \geq g2$ e $g2 \geq g1$, temos que $g1 = g2$. Podemos concluir então que a métrica $\mu_{sistema}$ apresenta as propriedades de ordem desejáveis a uma métrica.

4.6.3 Tipos de Escala

Essa propriedade preocupa-se em definir qual tipo de escala a métrica definida obedece. Existe um conjunto de tipos de escalas já apresentadas na Seção 3.2.3. Dentre os tipos de escala definidos, temos que aquele que mais se encaixa com $\mu_{sistema}$ é o tipo de escala de taxa, pois além de permitir ordenação e a distância entre seus valores ter significado, $\mu_{sistema}$ permite realizar análise percentual e operações aritméticas (visando observar o incremento percentual em relação à última avaliação). Esse tipo de escala é vista com bons olhos por diversos autores, que a indicam como a melhor opção para uma métrica [23].

4.6.4 Estruturas Extensivas

O sistema relacional formal **Progresso do Projeto** define um conjunto *Progresso do Projeto*, que contém o valores resultantes do cálculo do progresso funcional de vários sistemas. De modo a garantir que esse sistema relacional formal permita a realização de comparações e operações entre elementos de *Progresso do Projeto*, foram observadas as seguintes propriedades:

A1. Existem as relações $\geq, \leq$ sobre *Progresso do Projeto*, que consistem em relações de fraca ordem.

A2. Ela possui fraca associatividade para operações "+", "-" permitidas sobre o sistema relacional formal. Por exemplo, seja $g1, g2, g3 \in Progresso\ do\ Projeto$, temos que $g1 - (g2 - g3) = (g1 - g2) - g3$.

A3. A fraca comutatividade é observada, quando for usado fatores como escala, onde a operação "×" é utilizada. Por exemplo, seja *f* um valor inteiro a ser aplicado como escala sobre o progresso g, nesse caso, temos que $f \times g = g \times f$.

A4. A fraca monotonicidade é observada com as operações "×" e "+". Por exemplo, sejam $g1, g2, g3 \in Progresso\ do\ Projeto$, com $g1 \geq g2$, temos que $g1 + g3 \geq g2 + g3$, bem como $g3 + g1 \geq g3 + g2$.

A5. O Axioma de Arquimedes é válido para as relações “>”, “≥”, “<”, “≤”, e operações binárias “+”, “×”, “-“, “÷”. Por exemplo, para $g1, g2,$ $g3, g4 \in Progresso\ do\ Projeto$, e operação binária “+”, temos que, se $g3 > g4$, existe um valor natural n, tal que $g1 + n \times g3 > g2 + n \times g4$.

Assim, todos os axiomas relacionados a uma estrutura extensiva, ou seja, a uma métrica que permita a inserção de operações binárias, são obedecidos pelo sistema relacional formal definido pela métrica $\mu_{sistema}$.

4.6.5 Atomicidade

A métrica $\mu_{sistema}$ tem como objetivo principal recuperar informações sobre o progresso de projetos de software. A sensibilidade desta métrica, em relação a mudanças no projeto, é definida em termos dos casos de uso realizados (quais as etapas completadas e os artefatos produzidos). Esta sensibilidade indica o progresso atingido no desenvolvimento de um projeto, mas de um ponto de vista mais técnico, observando as funcionalidades do sistema em desenvolvimento. Existem, no entanto, outras propriedades que também indicam progresso como, por exemplo, o progresso das atividades planejadas e necessárias para o desenvolvimento. Tais propriedades representam aspectos de progresso que não são suportados pela métrica, e devem ser capturados de outra forma. Abaixo, seguem alguns exemplos de modificações atômicas e não atômicas para a métrica definida:

Mod1. Concluir um determinado artefato, relacionado ao desenvolvimento de um caso de uso do sistema.

Mod2. Identificar que uma determinada etapa, do caso de uso, foi concluída.

Mod3. Retirar um caso de uso do sistema (ainda não trabalhado), e inserir outro (também não trabalhado).

Mod4. Retirar um caso de uso *x* do sistema (parcialmente modelado), e inserir um caso de uso *y* (também parcialmente modelado).

Mod5. Produzir um artefato necessário para realização da etapa "Análise e Projeto" do caso de uso *x*, e verificar que, um outro artefato, dessa mesma etapa, que tinha sido considerado feito, ainda precisa de trabalho para ser realizado. Nesse exemplo, os dois artefatos possuem o mesmo peso associado.

Das quatro modificações acima, temos que Mod1 e Mod2 não consistem em modificações atômicas, pois acarretam em mudanças no progresso do sistema. Mod4 possivelmente representará uma mudança no progresso do projeto. Ela só não representará uma mudança, caso o produto do progresso do caso de uso *x* pelo seu peso associado ($p_x \times \mu_x$) coincidir com o produto do caso de uso *y* pelo seu peso associado ($p_y \times \mu_y$). Já Mod3 e Mod5 não representam nenhuma modificação em relação ao progresso atual do sistema.

4.7 Cobertura da Métrica em Relação às Métricas de Progresso Existentes

$\mu_{sistema}$ representa a tentativa de se obter uma métrica global, que identifica o progresso do sistema em termos percentuais, onde se pode observar, de maneira clara e simples, o quanto foi feito e o que ainda falta fazer. Desse modo, um dos principais objetivos de $\mu_{sistema}$ é incorporar a capacidade de avaliar as diversas propriedades que indicam progresso na funcionalidade do sistema. Como consequência dessa definição, temos que $\mu_{sistema}$ engloba propriedades que são consideradas em diversas métricas de avaliação de progresso existentes, citadas na Seção 3.5. A Tabela 4.3 representa um resumo do relacionamento de $\mu_{sistema}$ com tais métricas.

O que se pode observar é que $\mu_{sistema}$ preocupa-se com as propriedades que representam incrementos no projeto em termos de documentação e de funcionalidades inseridas no produto final. As métricas *Atividades Planejadas vs. Realizadas* e *Data dos Marcos de Referência* que não são cobertas por $\mu_{sistema}$, preocupam-se com o desempenho das equipes de desenvolvimento durante o projeto. Desse modo, observa-se

que, aspectos relacionados às equipes de desenvolvimento, à organização, e a outros fatores externos, não são cobertos por $\mu_{sistema}$. Visando superar esse tipo de defeiciência, temos que o processo de avaliação de progresso, apresentado no Capítulo 5, utiliza, além de $\mu_{sistema}$, um conjunto de métricas que avaliam o progresso, segundo aspectos de desempenho.

Métrica	Cobertura	Observações
Atividades Planejadas vs. Realizadas	Nenhuma	Não considera a realização das atividades, o esforço envolvido, nem fatores organizacionais que influenciam nessas atividades. Considera apenas, os artefatos resultantes da realização dessas atividades.
Status dos Componentes	Parcial	Considera se os componentes necessários para realização do caso de uso foram produzidos, mas não há nenhum esforço na tentativa de estimar o número de componentes que deverão existir em um instante *t*, ou quantificar a quantidade de componentes produzidos.
Status dos Requisitos	Total	O *status* de um requisito é considerado a partir da inspeção do caso de uso que representa esse requisito.
Status dos Casos de Teste	Total	Os casos de teste são considerados durante a avaliação dos testes relacionados a cada caso de uso.
Cenários Testados	Total	Verifica os cenários testado durante a avaliação dos teste do caso de uso.
Status do Relatório de Problemas	Parcial	Considera os testes que resultaram em erros identificados, mas não há nenhuma preocupação quanto ao número de defeitos que ainda precisam ser resolvidos ou já foram solucionados.
Número de Componentes do Build	Parcial	Preocupa-se com a existência dos componentes, mas não quantifica a quantidade de componentes relacionados com um *build*.
Funcionalidades do Build	Total	Existe a checagem das funcionalidades incorporadas no sistema, para cada caso de uso.
Número da iteração (para processo de desenvolvimento iterativo)	Total	$\mu_{sistema}$ contém um índice *i*, que representa a iteração em que o projeto se encontra.
Número de contratos completos	Parcial	É possível identificar quantos contratos foram completos, se identificando quais casos de uso pertencem a quais contratos.
Data dos marcos de referência	Nenhuma	$\mu_{sistema}$ não leva em consideração marcos de referência, nem as datas planejadas para que esses tenham sido atingidos.

Tabela 4.3. Cobertura de $\mu_{sistema}$ em relação às demais métricas de progresso

As métricas parcialmente cobertas não representam falha, pois não é escopo de $\mu_{sistema}$ quantificar os diversos elementos relacionados ao desenvolvimento do sistema.

Ao contrário, o objetivo é obter um valor único e global, que identifique quantitativamente o progresso do projeto, simplificando a análise do progresso, e minimizando a quantidade de dados que devem ser manipulados pelo gerente de projeto.

4.8 Limitações da Métrica de Progresso Funcional

Apesar de tentar ser a mais abrangente possível, a métrica de progresso funcional do projeto, apresenta algumas limitações que ainda precisam ser superadas:

- O exemplo de critérios de inspeção apresentados para avaliar o conjunto padrão de artefatos encontrados a partir da análise do RUP é limitado, capturando-se apenas as relações sintáticas entre os artefatos e não se preocupando em observar questões importantes como corretude (localização de defeitos), que devem ser inseridos quando a métrica for aplicada sobre projetos reais.
- A influência dos aspectos não funcionais não é considerada. Desse modo não são consideradas características de qualidade do produto, tais como: desempenho, usabilidade, disponibilidade, entre outros.
- Não consegue capturar a noção de desempenho das equipes de desenvolvimento, ou seja, não leva em consideração as atividades planejadas e realizadas, as datas dos marcos de referências, entre outros.
- A coleta das informações necessárias para se calcular o valor de $\mu_{sistema}$ é extensa e repetitiva, podendo desmotivar o coletor. Essa limitação vem do fato que $\mu_{sistema}$ deriva diretamente da inspeção dos vários artefatos de cada caso de uso relacionado ao projeto, o que pode resultar, em centenas de inspeções sobre artefatos, dependendo do tamanho do projeto. Assim, observa-se a necessidade da definição e construção de uma ferramenta que automatize o processo de inspeção dos artefatos, e calcule, a partir dos resultados das inspeções, o progresso funcional do sistema.

É importante definir esse conjunto de limitações como proposta para futuros trabalhos relacionados com a área de avaliação de progresso. Algumas dessas limitações são inerentes à definição da métrica, portanto, dificilmente serão eliminadas, porém

algumas melhorias poderão ser definidas visando automatizar a coleta, o cálculo da métrica e garantir a precisão dos resultados.

4.9 Considerações Finais

A métrica $\mu_{sistema}$ definida neste capítulo busca incorporar os principais aspectos envolvidos com o desenvolvimento de um projeto, capturando o progresso através da observação das funcionalidades que devem ser desenvolvidas no sistema. Tais funcionalidades são verificadas a partir da realização dos diversos casos de uso encontrados durante a definição do problema a ser resolvido. É essencial para a identificação do progresso funcional de um projeto, inspecionar os diversos artefatos que são necessários para realização dos casos de uso e, a partir dos resultados dessa inspeções e do esforço necessário para produção de cada artefato, $\mu_{sistema}$ é calculada.

O cálculo do progresso funcional envolve uma série de ponderações e cálculos que, embora não sejam complexos, são melhores esclarecidos através de exemplos que permitem observar as dificuldades inerentes à sua utilização. Assim, visando tornar mais clara a definição dos conjuntos envolvidos com a métrica e os cálculos que devem ser realizados, a aplicação de $\mu_{sistema}$ será exemplificada durante a definição do processo de avaliação de progresso (Seção 5.2.2) e no Capítulo 6, que apresenta o estudo de caso realizado para validar o processo definido.

O escopo da avaliação de progresso realizada pelo processo de avaliação definido neste trabalho pretende identificar todos os aspectos que representam, de alguma forma, progresso em um projeto de desenvolvimento de software. Desse modo, devido as limitações observadas em $\mu_{sistema}$, que não consegue recuperar informações sobre a produtividade e desempenho das equipes de desenvolvimento, bem como a qualidade no planejamento do projeto, foram definidas outras métricas, apresentadas na Seção 5.2.1, que também serão utilizadas no processo, resultando em uma visão mais precisa e completa do progresso de um determinado projeto.

Capítulo 5

Definição do Inspector

Este capítulo apresenta a definição de um processo para avaliação de progresso de projetos de software orientado a objetos que usam o conceito de casos de uso para representar as funcionalidades do sistema. O escopo do trabalho focaliza a definição de um processo, que utiliza um conjunto reduzido de métricas de progresso para mostrar o estado atual do projeto, visando apoiar o gerente de projeto na tomada de decisões, análise de desempenho e fornecer subsídios para que ele possa fornecer um relato preciso do estado do projeto ao cliente.

O Inspector, denominação dada ao processo, define um conjunto de atividades que buscam sistematizar o uso de técnicas e métricas para avaliação do progresso técnico de um projeto de desenvolvimento de software orientado a objetos. A aplicação deste processo é indicada para projetos de médio e grande porte, onde existe a necessidade de um gerenciamento mais próximo, capaz de identificar problemas e possíveis atrasos. Além disso, o processo visa garantir mecanismos que permitam fornecer relatórios, identificando o *status* do projeto sempre que o cliente desejar.

As subseções seguintes apresentam os principais conceitos introduzidos no Inspector, as duas visões de progresso fornecidas, os responsáveis envolvidos, a classificação dos artefatos que são utilizados durante a avaliação de progresso, o fluxo de atividades definido, algumas limitações existentes e considerações finais.

5.1 Conceitos

A terminologia adotada pelo Inspector é bastante similar àquela utilizada pela Rational para descrever o Processo de Desenvolvimento Unificado (RUP) [7]. O RUP representa um processo comercial, iterativo e incremental, que apresenta todas as características desejáveis para aplicação do Inspector (utiliza linguagem de modelagem UML e é dirigido através do desenvolvimento de casos de uso). Desse modo, o uso de uma notação semelhante à do RUP visa facilitar uma futura integração do Inspector à este processo de desenvolvimento, que resultaria no processo de gerenciamento do RUP com capacidade para o controle do progresso de projetos de software. Desse modo, assim como no RUP, os principais conceitos encontrados na definição do Inspector são: responsáveis, artefatos e atividades.

5.1.1 Responsáveis

Corresponde a um papel (ou posição) que pode ser atribuído a uma pessoa ou grupo, requer responsabilidade e ações, tais como realizar atividades e produzir artefatos [7]. Uma pessoa pode assumir o posto de mais de um responsável, ou seja, podemos ter um membro da equipe desempenhando mais de um papel durante a avaliação de progresso. Um exemplo seria o membro da equipe ser, ao mesmo tempo, Gerente de Projeto e Coletor das métricas durante a segunda avaliação de progresso.

5.1.2 Artefatos

Termo geral utilizado para qualquer tipo de informação criada, produzida, alterada ou utilizada pelos responsáveis no desenvolvimento do sistema [7]. Exemplos de artefatos são: diagramas UML e seus textos associados, código fonte, projetos de interface, protótipos, entre outros. Os artefatos, no Inspector, são classificados em três categorias distintas, de acordo com a sua participação no processo: artefatos produzidos, artefatos inspecionados e artefatos de apoio. A primeira representa os artefatos que devem ser gerados durante a monitoria e avaliação do progresso do projeto. A segunda representa os artefatos que devem ser inspecionados para identificar o progresso funcional dos casos de uso e do sistema, e o desempenho das equipes de

desenvolvimento. Esse segundo conjunto de artefatos deriva do processo de desenvolvimento utilizado, e pode variar de projeto para projeto. Já a terceira, representa os artefatos de planejamento e gerenciamento que são produzidos durante o desenvolvimento do projeto, e são usados como fonte de apoio ao planejamento da coleta das métricas e avaliação do progresso técnico. A Seção 5.4, apresenta uma descrição mais detalhada dos artefatos envolvidos com o Inspector.

5.1.3 Atividades

Uma atividade corresponde a uma unidade de trabalho tangível, coordenada por um responsável, em um fluxo de trabalho[10] que: atribui tarefas para cada responsável, produz um resultado bem definido (conjunto de artefatos) e representa uma unidade de trabalho com limites definidos em um plano de projeto, onde as tarefas são atribuídas aos indivíduos [7].

O Inspector define um conjunto de atividades, visando o acompanhamento de projetos:

- Avaliar o Status das Métricas na Organização;
- Adaptar o Inspector à Organização;
- Instanciar o Inspector;
- Planejar Avaliação do Progresso Técnico;
- Coletar e Processar Dados de Desempenho;
- Coletar e Processar Dados de Progresso Funcional;
- Avaliar Resultados; e
- Solucionar Problemas.

[10] Fluxo de trabalho corresponde a um conjunto de atividades agrupadas que pretendem cobrir algum aspecto do desenvolvimento de software.

Esse conjunto de atividades define o trabalho envolvido com a adaptação da organização, planejamento, coleta, cálculo e avaliação das métricas de progresso de um determinado projeto. Cada atividade é descrita, com mais detalhes, na Seção 5.5.

5.2 As Duas Visões de Progresso do Inspector

O Inspector apresenta um conjunto de métricas e artefatos que, quando coletados e analisados, fornecerão ao gerente de projeto duas visões complementares do progresso do sistema. São elas:

- visão de desempenho; e
- visão de funcionalidade.

A obtenção sistemática de ambas, permite ao gerente de projeto acompanhar e monitorar o desenvolvimento do projeto, analisar o desempenho das equipes envolvidas, verificando a necessidade de se realizar deslocamentos no cronograma ou alocar pessoal para o projeto e localizar, a nível de projeto, e até mesmo de caso de uso, onde estão ocorrendo possíveis problemas.

5.2.1 Desempenho

Tal perspectiva focaliza a realização das atividades previamente planejadas, para cada equipe envolvida no projeto, um período de tempo previsto. A principal preocupação dessa visão é fornecer informações suficientes ao gerente de projeto, para permitir que o mesmo encontre possíveis atrasos no cronograma e qual a influência deste atraso no restante do projeto. Além disso, a partir dessas informações, pode-se verificar a necessidade de inserir, realocar ou retirar pessoal do projeto, visando superar as dificuldades encontradas.

A visão de desempenho recupera os gráficos Gantt (Figura 3.3), inicialmente criados, contendo as previsões iniciais para cada atividade: data de início, data de término e responsáveis. O modo de se realizar tais previsões não faz parte do escopo do Inspector, sendo que [30, 44] apresentam algumas técnicas que podem ser utilizadas para fazer estas estimativas. Além disso, a visão de desempenho recupera também os

gráficos Gantt que representam a situação atual do projeto, indicando a data de início real da atividade, a data de término (se tiver terminado) ou a porcentagem da atividade que foi concluída. Através da comparação destes gráficos são identificadas possíveis equipes em dificuldades e atividades em atraso ou muito adiantadas.

Às vezes, atraso em uma determinada atividade não implica necessariamente em atraso no projeto [29]. Afim de controlar e melhor visualizar o impacto do atraso de uma determinada atividade no projeto, a técnica PERT (*Program Evaluation and Review Technique*) [5] é utilizada. Tal técnica consiste na construção e constante atualização de um gráfico, denominado gráfico PERT, que correspondem a um grafo, onde os nós são as atividades listadas no gráfico Gantt e as arestas são as dependências entre tais atividades (Figura 5.1). Tal gráfico insere a noção de caminho crítico, que indica quais atividades são críticas ao cronograma do projeto. O caminho crítico corresponde ao caminho de atividades mais longo, ou seja, que leva mais tempo para a realização do projeto [5]. Na Figura 5.1 o caminho crítico corresponde às atividades 1, 2, 5 e 6. Atividades críticas não podem atrasar, pois acarretam em atrasos globais ao projeto. Desta forma, o gerente de projeto deve desprender uma atenção particular para cada uma dessas atividades do caminho crítico, de modo a minimizar o atraso no projeto, e evitar que problemas nessas atividades venham a ocorrer.

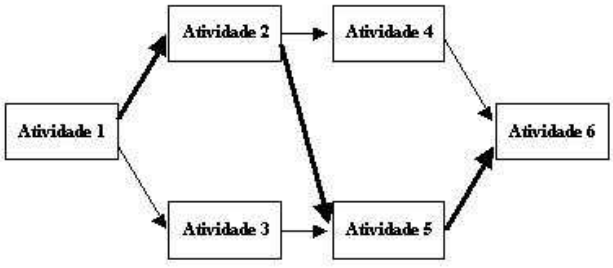


Figura 5.1. Gráfico PERT simplificado mostrando o caminho crítico.

A utilização sistemática de um processo de acompanhamento das atividades, que se baseia na manipulação e atualização de tais gráficos (PERT e Gantt de atividades), permite ao gerente de projeto uma melhor visão do desempenho de uma determinada equipe de desenvolvimento. De posse destes gráficos, é possível ao gerente de projeto identificar situações onde o atraso é inevitável, prevenir que pequenos atrasos no projeto

alcancem proporções maiores, utilizando medidas de prevenção mais adequadas à situação.

Além do acompanhamento das atividades, é interessante possuir alguma métrica que indique quantitativamente o desempenho das equipes de desenvolvimento. O Inspector define três métricas que auxiliam na avaliação do desempenho das equipes de desenvolvimento. São elas: $\mu_{concluído}$, $\mu_{atraso\ médio}$ e $\mu_{novas\ atividades}$.

A métrica $\mu_{concluído}$ representa o quanto uma determinada equipe já concluiu das atividades que estavam inicialmente planejadas para terminarem dentro do escopo de tempo da avaliação. Desse modo, temos que $\mu_{concluído}$ é dada pela equação:

$$\mu_{concluído}(equipe) = \frac{\sum_{x \in Atividades(equipe)} tempoEstimado(x) \times concluído(x)}{\sum_{x \in Atividades(equipe)} tempoEstimado(x)} \quad (5.1)$$

onde *equipe* indica a equipe que está sendo avaliada, *Atividades*(*equipe*) representa o conjunto de atividades relacionadas com *equipe* que estavam inicialmente planejadas para serem concluídas no escopo de tempo da avaliação, *concluído*(x) consiste na função que retorna o quanto da atividade x se realizou, com $0 \leq concluído(x) \leq 1$, e *tempoEstimado*($x$) é a função que retorna o tempo estimado para a atividade x, ou seja, é a função que indica o quanto se espera que a atividade x demore para ser realizada. Caso a atividade tenha sido completamente realizada, temos que $concluído(x) = 1$, ou seja 100%, e *tempoEstimado*(x) é igual ao tempo total gasto na realização da atividade. Caso contrário, a pessoa responsável pela coleta deverá, através de conversa com o responsável pela atividade, estimar o quanto se foi concluído e qual o tempo estimado para se concluir a mesma.

A métrica $\mu_{atraso\ médio}$ indica, quantitativamente, o percentual de atraso médio por atividade que está sendo enfrentado por uma determinada equipe de desenvolvimento. Desse modo, temos que $\mu_{atraso\ médio}$ é dada pela equação:

$$\mu_{atraso\ médio}(equipe) = \frac{\sum_{x \in\ Atividades(equipe)} \frac{tempoEstimado_f(x) - tempoEstimado_i(x)}{tempoEstimado_i(x)}}{\#\,Atividades(equipe)} \quad (5.2)$$

onde *equipe* indica a equipe que está sendo avaliada, *Atividades*(*equipe*) representa o conjunto de atividades relacionadas com *equipe*, que estavam inicialmente planejadas para serem concluídas no escopo de tempo da avaliação, $tempoEstimado_i(x)$ é a função que retorna o tempo inicialmente estimado para a atividade x, $tempoEstimado_f(x)$ é a função que retorna o tempo gasto para realização da atividade x, caso ela tenha sido completada, ou, caso contrário, retorna a estimativa do novo tempo necessário para realização desta atividade, e $\#\,Atividades(equipe)$ é a cardinalidade de $Atividades(equipe)$.

O valor de $tempoEstimado_f(x)$ da Equação 5.2, caso a atividade não tenha sido completamente realizada, é retirado a partir de uma regra de três simples:

$$tempoEstimado_f(x) = \frac{tempoGasto(x)}{concluído(x)} \quad (5.3)$$

onde $tempoGasto(x)$ é a função que retorna a quantidade de tempo já trabalhado na realização da atividade x, e $concluído(x)$ representa a função que indica a porcentagem da atividade que foi concluída nestes dias trabalhados.

O escopo da avaliação define o intervalo de tempo que a avaliação de progresso irá considerar, consequentemente, serão analisadas as atividades que estavam planejadas para serem executadas dentro desse intervalo de tempo. Porém, muitas vezes, surgem novas atividades, não planejadas, que alocam o pessoal da equipe e, dessa forma, prejudica o planejamento inicial e o desempenho. O cálculo da métrica $\mu_{novas\ atividades}$ tem como objetivo indicar a porcentagem média de tempo gasto com novas atividades que surgem dentro do escopo da avaliação. Assim sendo, valores altos para $\mu_{novas\ atividades}$ representam problemas de planejamento que aumentam os riscos no desenvolvimento

do projeto e podem indicar problemas graves no desempenho das equipes. Esta métrica é calculada através da seguinte equação:

$$\mu_{novas\ atividades} = \frac{\sum_{x \in\ Atividades\ Iniciadas} tempoGasto(x) - \sum_{x \in\ (AtividadesIniciadas\ \cap\ AtividadesPlanejadas)} tempoGasto(x)}{\sum_{x \in\ AtividadesIniciadas} tempoGasto(x)} \quad (5.4)$$

onde *AtividadesIniciadas* é o conjunto que contém as atividades realmente iniciadas dentro do escopo da avaliação, *AtividadesPlanejadas* é o conjunto que contém as atividades planejadas para serem iniciadas dentro desse escopo, e *tempoGasto*(x) é a função que retorna a quantidade de tempo já trabalhado para realização da atividade x. Desse modo, a métrica identifica o tempo gasto com atividades que foram iniciadas, ou seja, que pertencem ao conjunto *AtividadesIniciadas*, mas não foram planejadas, ou seja, não pertencem ao conjunto *AtividadesPlanejadas* e, em seguida, calcula o percentual que estas atividades representam em relação ao total de atividades iniciadas.

Desse modo, o acompanhamento sistemático das atividades, através dos cronogramas de atividades e gráficos PERT, juntamente com o cálculo e análise das métricas $\mu_{concluído}$, $\mu_{atraso\ médio}$ e $\mu_{novas\ atividades}$, fornece ao gerente uma visão precisa do desempenho das equipes de desenvolvimento. Equipes que apresentarem atrasos relativamente grandes e dificuldades em realizar determinada atividade, devem ser consultadas a fim de se identificar precisamente o problema, e para que soluções possam ser encontradas.

A visão de desempenho focaliza essencialmente a avaliação das equipes envolvidas com o projeto, observando as atividades planejadas, verificando o *status* das mesmas e identificando a realização destas atividades pela equipe. Ao contrário da visão de funcionalidade, a visão de desempenho independe do paradigma de desenvolvimento utilizado (estruturado, orientado a objetos, funcional, etc.). Além disso, ela não cobre os aspectos funcionais do projeto, identificando o quanto das funcionalidades planejadas para o sistema já foram incorporadas. Visando observar esses aspectos e identificar

problemas ocorridos durante o desenvolvimento dessas funcionalidades, temos que a visão de funcionalidade foi definida (Seção 5.2.2).

Exemplo de obtenção da visão de desempenho

A obtenção da visão de desempenho exige a recuperação dos cronogramas de atividades, que servem como fonte de informação para o cálculo das métricas de desempenho definidas. A Tabela 5.1 mostra um cronograma de atividades resumido de um projeto X em uma determinada data A.

Id	Nome da Atividade	Duração Estimada	Concluído (%)
1	Especificação do módulo de compras	5 hs	0%
2	Definição da arquitetura do módulo de compras	14 hs	0%
3	Modelagem dos dados do módulo de compras	12 hs	0%
4	Implementação	20 hs	0%
5	Testes de integração da comunicação	4 hs	0%
6	Testes de uso do módulo de compras	3 hs	0%

Tabela 5.1. Cronograma de atividades de um projeto X em uma data A

Alguns dias depois, em uma data B, foi gerado um cronograma atualizado para o projeto X, mostrado na Tabela 5.2. De posse do cronograma atualizado (produzido na data B) e do cronograma anterior (gerado na data A), é possível calcular o desempenho da equipe de desenvolvimento nesse intervalo de tempo.

Id	Nome da Atividade	Duração Estimada	Concluído (%)
1	Especificação do módulo de compras	7 hs	100%
2	Definição da arquitetura do módulo de compras	17 hs	100%
3	Correção de falhas no módulo de vendas	13 hs	100%
4	Modelagem dos dados do módulo de compras	15 hs	90%
5	Implementação	25 hs	20%
6	Participação em cursos de capacitação	12 hs	100%
7	Testes de integração da comunicação	5 hs	25%
8	Testes de uso do módulo de compras	3 hs	0%

Tabela 5.2. Cronograma de atividades de um projeto X em uma data B

O cálculo, para uma determinada equipe, do quanto foi concluído das atividades que estavam inicialmente planejadas para terminarem no escopo da avaliação, ou seja, o cálculo de $\mu_{concluído}$ é simples e feito a partir do cronograma atualizado, observando a coluna que indica porcentagem de conclusão das atividades na data da avaliação e o tempo estimado de conclusão de cada atividade. A equação 5.1, que indica o cálculo dessa métrica, é mostrada novamente abaixo:

$$\mu_{concluído}(projeto\ X) = \frac{\sum_{x\ \in\ Atividades(projeto\ X)} tempoEstimado(x) \times concluído(x)}{\sum_{x\ \in\ Atividades(projeto\ X)} tempoEstimado(x)}$$

Assim, para a data B, $\mu_{concluído}$ apresenta o seguinte resultado:

$$\mu_{concluído}(projeto\ X) = \frac{7\times1+17\times1+13\times1+15\times0{,}9+25\times0{,}2+12\times1+5\times0{,}25+3\times0}{7+17+13+15+25+12+5+3} = \frac{68{,}75}{97} = 0{,}71$$

O valor obtido indica que foram concluídas 71% das atividades pertencentes ao escopo da avaliação. O valor ideal para essa métrica seria 100% das atividades concluídas, valores menores indicam que algumas das atividades apresentaram algum tipo de atraso.

O atraso médio por atividade, ou seja, o cálculo de $\mu_{atraso\ médio}$ é feito a partir da comparação da duração estimada das atividades no cronograma atual com a duração estimada destas atividades no cronograma produzido anteriormente. Esta métrica é calculada segundo a Equação 5.2 mostrada novamente abaixo:

$$\mu_{atraso\ médio}(equipe) = \frac{\sum_{x\ \in\ Atividades(equipe)} \frac{tempoEstimado_f(x) - tempoEstimado_i(x)}{tempoEstimado_i(x)}}{\#\,Atividades(equipe)}$$

Substituindo os dados do exemplo nesta equação temos:

$$\mu_{atraso\ médio}(equipe) = \frac{\frac{7-5}{5}+\frac{17-14}{14}+\frac{15-12}{12}+\frac{25-20}{20}+\frac{5-4}{4}+\frac{3-3}{3}}{6} = 0{,}23$$

Neste exemplo, o atraso médio por atividade é de 23%, independente do tamanho da atividade. O valor ideal para essa métrica se aproxima de 0%, indicando que não há nenhum atraso durante a realização das atividades planejadas.

Por fim, a qualidade no planejamento é observada através do surgimento de novas atividades que podem prejudicar o desempenho da equipe de desenvolvimento. Essa

propriedade é calculada através da métrica $\mu_{novas\ atividades}$, que corresponde à Equação 5.4 e é mostrada novamente a seguir:

$$\mu_{novas\ atividades} = \frac{\sum_{x\ \in\ AtividadesIniciadas} tempoGasto(x) - \sum_{x\ \in\ (AtividadesIniciadas\ \cap\ AtividadesPlanejadas)} tempoGasto(x)}{\sum_{x\ \in\ AtividadesIniciadas} tempoGasto(x)}$$

Colocando os dados do exemplo nesta equação, temos o seguinte tempo gasto com todas as atividades iniciadas e com as atividades que não foram planejadas:

$$\sum_{x\ \in\ AtividadesIniciadas} tempoGasto(x) = 7\times1+17\times1+13\times1+15\times0{,}9+25\times0{,}2+12\times1+5\times0{,}25+3\times0 = 68{,}75$$

$$\sum_{x\ \in\ AtividadesIniciadas\ \cap\ AtividadesPlanejadas} tempoGasto(x) = 7\times1+17\times1+15\times0{,}9+25\times0{,}2+5\times0{,}25+3\times0 = 43{,}75$$

Assim, o valor de $\mu_{novas\ atividades}$ é:

$$\mu_{novas\ atividades} = \frac{68{,}75-43{,}75}{68{,}75} = 0{,}36$$

Desse modo, a quantidade de atividades não planejadas que tiveram de ser realizadas dentro do escopo do exemplo se aproxima de 36%, indicando um esforço significativo das equipes de desenvolvimento em atividades que surgiram sem planejamento. O valor ideal para essa métrica se aproxima de 0%, indicando que o pessoal está totalmente alocado às atividades anteriormente planejadas.

5.2.2 Funcionalidade

A visão de funcionalidade define o *status* do desenvolvimento das funcionalidades previstas para o sistema, indicando o aumento percentual no desenvolvimento das mesmas. O objetivo aqui é calcular o valor de $\mu_{sistema}$, ou seja, o progresso funcional do sistema, a partir do cálculo do progresso de cada caso de uso. No Capítulo 4 foi apresentada a descrição completa desta métrica, com todas as suas propriedades e implicações. A visão de funcionalidade, portanto, irá permitir a visualização do *status*

do projeto, bem como de cada caso de uso do sistema, identificando, em detalhes, as principais dificuldades técnicas apresentadas pelos membros das equipes durante o desenvolvimento.

A partir da análise da Tabela Resumo, gerada através do cálculo de $\mu_{sistema}$, poderá ser avaliada a situação atual de todos os casos de uso e do sistema. A Tabela Resumo permite identificar o progresso de cada etapa dos casos de uso, observando pontos onde não tem havido progresso. Um exemplo desta tabela é ilustrado na Tabela 4.2. Outro artefato gerado a partir do cálculo de $\mu_{sistema}$ são os gráficos de linha, indicando o progresso do sistema e de cada caso de uso, em relação ao tempo de desenvolvimento. A representação gráfica facilita a visualização do progresso e permite uma melhor análise de tendências.

Essa visão captura a construção do sistema a partir dos casos de uso, identificando todos os produtos que estão sendo gerados durante o desenvolvimento do sistema. Desse modo, ela é uma visão eficiente para se identificar problemas mais técnicos, voltados para definição e construção dos artefatos do projeto, bem como capturar pequenas deficiências no desenvolvimento, que dificilmente seriam recuperadas na visão de desempenho. Como o próprio nome diz, o foco dessa visão, são as funcionalidades do sistema, deixando a desejar na observação e captura dos aspectos não funcionais ou de qualidade, como fatores que também indicam progresso.

Exemplo de obtenção da visão de funcionalidade

A obtenção da visão de funcionalidade baseia-se no cálculo do progresso funcional do projeto em um determinado momento. O exemplo a seguir mostra como encontrar o valor de $\mu_{sistema}$ para um projeto X, cujo diagrama de caso de uso foi mostrado na Figura 4.1. Primeiramente, deverão ser identificados os casos de uso a serem avaliados. No exemplo, os casos de uso são:

- Cadastrar Tabelas;
- Atualizar Queixa;
- Consultar Informações; e

- Especificar Queixa.

Em seguida, os casos de uso identificados deverão ser analisados e, a partir dessa análise, deverão ser atribuídos pesos aos mesmos para que $\mu_{sistema}$ possa ser calculado. Suponhamos que no exemplo, os casos de uso tiveram a complexidade e o tamanho avaliados e foram-lhe atribuídos os seguintes pesos:

- $\mu_{Cadastrar\ Tabelas} = 2$;
- $\mu_{Atualizar\ Queixa} = 1$;
- $\mu_{Consultar\ Informações} = 1$; e
- $\mu_{Especificar\ Queixa} = 3$.

O próximo passo é identificar, para cada caso de uso, as etapas necessárias e os artefatos que deverão ser produzidos no desenvolvimento. Para simplificar o exemplo, assumiu-se que todos os casos de uso do projeto X passarão pelas mesmas etapas de desenvolvimento: $ETAPAS = \{análise,\ projeto,\ implementação\}$. Além disso, foi assumido que todas estas etapas representam mesmo peso no desenvolvimento do caso de uso, ou seja, $j \in ETAPAS \rightarrow p_j = 1$. Suponha também que, após uma análise dos casos de uso, ficou constatada a necessidade de produzir ou atualizar os seguintes artefatos:

- $A_{análise} = \{documento\ de\ requisitos\}$;
- $A_{projeto} = \{diagrama\ de\ classes,\ descrição\ dos\ dados\}$; e
- $A_{implementação} = \{código\ fonte,\ testes\ de\ unidade\}$.

Definidos os artefatos, deverão ser considerados os pesos dos mesmos para a produção de cada caso de uso. Para simplificar o exemplo, assumiu-se que todos os artefatos possuem peso 1, ou seja, $\forall x \in A_j, j \in ETAPAS \rightarrow p_x = 1$. O passo seguinte é inspecionar os artefatos definidos para cada caso de uso, obedecendo aos critérios de inspeção já apresentados na Seção 4.3. De acordo com o resultado da inspeção, valores

são atribuídos a $\mu_{caso\ de\ uso}(x)$, onde $x \in A_j$, $j \in ETAPAS$. Suponha que após a inspeção dos artefatos do exemplo, foram obtidos os seguintes valores para os casos de uso:

Cadastrar Tabelas

- $\mu_{análise}(Cadastrar\ Tabelas) = 1$
- $\mu_{projeto}(Cadastrar\ Tabelas) = 0{,}75$
- $\mu_{implementação}(Cadastrar\ Tabelas) = 0{,}15$

Assim, o progresso funcional do caso de uso Cadastrar Tabelas pode ser facilmente calculado a partir da Equação 4.6:

$$\mu_{Cadastrar\ Tabelas} = \frac{1\times 1 + 1\times 0{,}75 + 1\times 0{,}15}{3} = 0{,}63$$

Repare que, por não ser um fator tão decisivo para o exemplo, não está sendo considerada a iteração na qual a avaliação está sendo realizada.

Atualizar Queixa

- $\mu_{análise}(Atualizar\ Queixa) = 1$
- $\mu_{projeto}(Atualizar\ Queixa) = 0{,}7$
- $\mu_{implementação}(Atualizar\ Queixa) = 0{,}1$

Assim, o progresso funcional do caso de uso Atualizar Queixa pode ser facilmente calculado a partir da Equação 4.6:

$$\mu_{Atualizar\ Queixa} = \frac{1\times 1 + 1\times 0{,}7 + 1\times 0{,}1}{3} = 0{,}6$$

Consultar Informações

- $\mu_{análise}(Consultar\ Informações) = 0{,}8$
- $\mu_{projeto}(Consultar\ Informações) = 0{,}4$

- $\mu_{implementação}(Consultar\ Informações) = 0$

Assim, o progresso funcional do caso de uso Consultar Informações pode ser facilmente calculado a partir da Equação 4.6:

$$\mu_{Cosultar\ Informações} = \frac{1\times 0{,}8 + 1\times 0{,}4 + 1\times 0}{3} = 0{,}4$$

Especificar Queixa

- $\mu_{análise}(Especificar\ Queixa) = 1$
- $\mu_{projeto}(Especificar\ Queixa) = 1$
- $\mu_{implementação}(Especificar\ Queixa) = 0$

Assim, o progresso funcional do caso de uso Especificar Queixa pode ser facilmente calculado a partir da Equação 4.6:

$$\mu_{Especificar\ Queixa} = \frac{1\times 1 + 1\times 1 + 1\times 0}{3} = 0{,}66$$

Calculado o progresso funcional de cada caso de uso, o progresso funcional do projeto deriva diretamente da substituição destes valores na Equação 4.4, levando-se em consideração os pesos de cada caso de uso já atribuídos anteriormente. Desse modo tem-se:

$$\mu_{sistema} = \frac{2\times 0{,}63 + 1\times 0{,}6 + 1\times 0{,}4 + 3\times 0{,}66}{7} = 0{,}61$$

Esse resultado indica que 61% das funcionalidades previstas para o projeto X já foram incorporadas. De posse deste resultado, o gerente de projeto pode compará-lo com valores obtidos em avaliações anteriores e observar tendências sobre o projeto. Além disso, o progresso funcional de cada caso de uso também é útil, pois permite identificar funcionalidades específicas com problemas e atrasos no desenvolvimento.

5.2.3 Utilizando as Duas Visões

Como dito anteriormente, as duas visões de progresso apresentam informações complementares. Desse modo, a recuperação sistemática de ambas permitirá ao gerente obter uma representação mais precisa da situação do projeto, focalizando mais os aspectos relevantes a cada equipe de desenvolvimento (visão de desempenho), e observando o desenvolvimento do sistema sob o ponto de vista dos artefatos produzidos para realização dos casos de uso (visão de funcionalidade).

O Inspector define um conjunto de atividades, contendo passos que indicam o caminho para se realizar a atividade, visando fornecer maturidade suficiente à organização para que ela seja capaz de planejar, coletar, calcular e avaliar as duas visões apresentadas, sempre que for necessário, identificando problemas e buscando soluções para os mesmos. Estas atividades são apresentadas na Seção 5.5.

5.3 Responsáveis no Inspector

O Inspector define três responsáveis relacionados com a avaliação do progresso de um determinado projeto. Isso não implica, no entanto, que eles sejam os únicos envolvidos durante uma avaliação de progresso. Durante a realização das atividades, os responsáveis podem, e muitas vezes devem, envolver os demais membros da equipe ou *stakeholders* do projeto, a fim de realizar os passos necessários. Assim sendo, foram definidos os seguintes responsáveis:

Engenheiro de Processo

Responsável pelos diversos processos e técnicas utilizados pela organização (processo de desenvolvimento, processo de utilização de métricas, padrões de documentação, etc.). Suas responsabilidades incluem: configurar o processo antes do início do projeto, além de continuamente buscar técnicas e métodos para aperfeiçoar o processo durante o desenvolvimento. Visa fazer com que a organização atinja um maior nível de maturidade na construção de software.

No Inspector, ele é responsável por observar como se encontra atualmente a organização, inserir uma cultura de utilização das métricas de progresso definidas no Inspector dentro da organização e, instanciar as atividades e métricas do Inspector para um projeto específico.

Gerente de Projeto

Responsável por garantir que um determinado projeto da organização, seja realizado no tempo previsto e com a qualidade exigida tanto pelo cliente, como pelos padrões da organização [27]. Realiza o planejamento do projeto, com estimativas de custo, tempo e recursos necessários, além de fazer um acompanhamento constante, visando identificar possíveis dificuldades e buscando soluções para as mesmas.

No Inspector, o gerente de projeto é responsável pelo planejamento da avaliação do progresso técnico, identificando o escopo a ser considerado durante a avaliação. Ele é responsável pela análise dos valores obtidos com a coleta das métricas de desempenho e progresso funcional, comparando o atual com o estimado e analisando tendências. Além disso, o gerente deve encontrar soluções alternativas e aplicar a melhor solução para os problemas identificados.

Coletor de Informações

Responsável pela coleta de informações e métricas definidas no processo. Faz a recuperação das informações de desempenho das equipes no projeto, capturando os gráficos Gantt de atividades das diversas equipes de desenvolvimento, contendo o tempo estimado para realização das tarefas, e os Gantt atuais contendo a situação atual do projeto em função das atividades previstas. Outro artefato que o Coletor deve recuperar para avaliação do desempenho é o gráfico PERT contendo o caminho crítico de atividades que indica as atividades que não podem atrasar. Ainda relacionado com o desempenho, a partir da análise dos gráficos Gantt atuais e estimados, o Coletor calcula o quanto se concluiu das atividades previstas ($\mu_{concluído}$), o atraso médio percentual da equipe por atividade ($\mu_{atraso\ médio}$), e o surgimento de atividades não planejadas ($\mu_{novas\ atividades}$).

Durante a captura das funcionalidades adquiridas pelo sistema, o coletor recupera e calcula as métricas relacionadas ao progresso funcional do sistema, resultando na métrica $\mu_{sistema}$. Capturando e inspecionando os artefatos produzidos durante o desenvolvimento, o coletor identifica o *status* dos casos de uso do sistema, e consequentemente, o *status* do sistema como um todo. Os artefatos que devem ser inspecionados são definidos no início do projeto, juntamente com os critérios de inspeção que serão utilizados para avaliar se eles incorporam ou não a funcionalidade de um determinado caso de uso.

5.4 Artefatos do Inspector

O Inspector define um conjunto de artefatos, classificados em três categorias distintas, de acordo com sua participação no processo: artefatos produzidos, artefatos inspecionados e artefatos de apoio.

5.4.1 Artefatos Produzidos

Os artefatos produzidos referem-se aos artefatos que foram definidos no Inspector, e deverão ser produzidos durante a avaliação de progresso, visando planejar a avaliação, coletar, calcular e avaliar as métricas de progresso, relatar problemas e identificar soluções para os problemas encontrados. Na versão estendida do Inspector [37], foram definidos modelos para todos os artefatos dessa categoria, que servirão como padrão para documentação da avaliação de progresso. Tais modelos dos artefatos produzidos são apresentados com maiores detalhes no Apêndice A. Durante a aplicação do Inspector, os seguintes artefatos devem ser produzidos:

Visão Geral das Métricas na Organização

Descreve o estado atual da organização em termos do processo de desenvolvimento e ferramentas utilizadas, da competência e atitude das pessoas envolvidas, do perfil dos seus clientes e competidores, além da análise dos problemas que ela enfrenta para incorporação do conjunto de métricas definidas pelo Inspector. Este documento é usado pelo Engenheiro de Processo como uma base para adaptar o uso do Inspector na

organização, e configurá-lo para um projeto particular. Ele também é usado para explicar aos clientes porque é preciso controlar o desenvolvimento, criando motivação e um entendimento comum entre os membros da organização que são diretamente ou indiretamente afetados.

Critérios de Avaliação dos Artefatos

Este documento indica, para um projeto específico, o conjunto de possíveis etapas necessárias para o desenvolvimento de um caso de uso do sistema em construção, e quais artefatos poderão ser produzidos em cada uma dessas etapas. Além disso, para cada artefato identificado, deverá ser definido um conjunto de critérios, que servirão como fonte de consulta, durante a avaliação do progresso funcional (cálculo de $\mu_{sistema}$). Tais critérios irão indicar se um determinado artefato foi construído ou alterado, de forma a conter as informações necessárias para desenvolvimento de um determinado caso de uso.

Plano de Avaliação do Progresso Técnico

Documento que mantém informações sobre a avaliação de progresso a ser realizada. Define o escopo e a data da próxima avaliação de progresso, os responsáveis pela coleta das informações e quais as metas a serem atingidas na avaliação.

Modelo de Coleta de Informação sobre o Progresso do Projeto

Relatório que contém as informações necessárias para avaliação de progresso do projeto como um todo. Inicialmente, apresenta a identificação do projeto e do responsável pela coleta das informações contidas no modelo. Identifica quais são os casos de uso relacionados ao projeto, identificando o *status* de cada um no momento da avaliação ($\mu_{casodeuso}$). Possui também o campo, onde será preenchido o valor global da métrica de avaliação de progresso ($\mu_{sistema}$), e o incremento percentual em relação à avaliação anterior. Deriva diretamente dos valores obtidos na coleta de informações sobre o caso de uso, e serve como fonte de informação para avaliação do *status* funcional do projeto.

Modelo de Coleta de Informação sobre o Progresso do Caso de Uso

Esse modelo varia de projeto para projeto, sendo definido a partir do artefato Critérios de Avaliação dos Artefatos. Contém os artefatos que deverão ser inspecionados para um projeto específico e campos, onde os resultados da inspeção serão representados como forma de progresso percentual para o caso de uso. Os artefatos são subdivididos de acordo com a etapa do desenvolvimento do caso de uso em que eles são produzidos. Nem todos os casos de uso do projeto, obrigatoriamente, terão que desenvolver ou atualizar todos os artefatos definidos inicialmente. Desse modo, o modelo deve conter uma opção que permita ao Coletor de Informações assinalar que um determinado artefato não é obrigatório para o caso de uso em questão.

Avaliação do Progresso Técnico

Corresponde ao documento que representa as informações, resultantes da análise dos dados obtidos na coleta das informações sobre o desempenho das equipes e o progresso funcional do sistema. Nele são documentados os resultados observados, as tendências para o projeto, além dos problemas que foram identificados durante a avaliação. Composto de diversos outros artefatos, como: gráficos Gantt e PERT para análise do desempenho, tabela resumo contendo o *status* dos casos de uso e gráficos de linha para avaliação do progresso funcional, além da tabela que contém os problemas identificados durante a avaliação.

Documento de Solução dos Problemas Identificados

Corresponde ao documento que apresentará as soluções definidas pelo gerente de projeto, para cada problema identificado no relatório de problemas. Tal documento deve conter informações sobre os problemas encontrados, apresentar soluções para o problema, identificar a solução mais adequada e indicar a situação desse problema, após a aplicação da solução escolhida.

5.4.2 Artefatos Inspecionados

Os artefatos inspecionados, como o próprio nome diz, são aqueles artefatos que devem ser localizados e inspecionados durante a avaliação do progresso técnico do

projeto. Tais artefatos variam de acordo com o processo de desenvolvimento utilizado na organização, que define o conjunto de artefatos que deverão ser produzidos para o desenvolvimento de um determinado projeto. Estes artefatos, bem como os critérios de avaliação que deverão ser observados durante a inspeção dos mesmos, são especificados no artefato Critérios de Avaliação dos Artefatos, durante a realização da atividade Instanciar o Inspector.

A Seção 4.3 definiu um conjunto padrão de artefatos que devem ser inspecionados, baseado em um estudo realizado sobre o RUP [7]. Dessa forma, organizações que utilizem o RUP como processo de desenvolvimento de software, podem configurar $\mu_{sistema}$ com o conjunto de artefatos apresentado, e utilizar seus respectivos critérios de avaliação. Nada impede que novos artefatos sejam inseridos nesse conjunto, ou que artefatos citados não necessitem ser produzidos e sejam removidos. É interessante manter um documento que contenha todos os artefatos que podem ser produzidos na organização, independente do projeto que vai ser desenvolvido, servindo, desse modo, como fonte de consulta durante a identificação dos artefatos necessários para um projeto.

5.4.3 Artefatos de Apoio

Correspondem aos artefatos que o Inspector assume como sendo produzidos pela organização durante o desenvolvimento do projeto, e são utilizados como fonte de entrada, auxiliando no planejamento da avaliação do progresso técnico e análise dos resultados obtidos. O Inspector classifica os seguintes artefatos como sendo de apoio:

Glossário da Organização

Representa um dicionário de dados contendo as principais notações e conceitos necessários aos negócios com o qual a organização está envolvida. No Inspector, este artefato é útil para o Engenheiro de Processo se familiarizar com os conceitos necessários para o desenvolvimento dos sistemas que a organização é responsável, e identificar os principais negócios que ela está envolvida.

Documentação do Processo de Desenvolvimento

Consiste no documento que contém a representação textual do processo de desenvolvimento utilizado pela organização. Representa uma excelente fonte de informação para o Engenheiro de Processo obter uma visão geral do processo de desenvolvimento, e verificar se a organização utiliza algum tipo de métrica para monitorar o desenvolvimento e a qualidade do produto que está sendo desenvolvido.

Plano de Projeto

Este artefato contém as informações relativas ao planejamento do projeto, alocando recursos, definindo escopo e atividades necessárias para o desenvolvimento. Esse plano servirá como fonte de informação para identificar quais são as metas esperadas para a avaliação de progresso a ser realizada. Além disso, ele contém os gráficos Gantt estimados, devendo ser consultados para recuperação de tais informações.

Plano de Iteração

Esse artefato é produzido somente quando o processo de desenvolvimento for iterativo, por exemplo, o processo unificado (RUP). Nesse caso, é interessante produzir um documento de auxílio ao gerenciamento, planejando as atividades de desenvolvimento que deverão ser realizadas em uma determinada iteração. Esse plano terá função semelhante ao plano de projeto no Inspector, servindo como fonte de consulta para o planejamento da avaliação do progresso técnico e recuperação das informações de desempenho.

5.5 O Fluxo de Atividades do Inspector

Essa seção apresenta uma descrição das atividades definidas no Inspector, que visam fornecer maturidade suficiente à organização para captura, cálculo e análise das métricas de progresso definidas. A Figura 5.2, mostra o fluxo de atividades, contendo as atividades, a dependência de execução entre elas e o responsável por cada atividade.

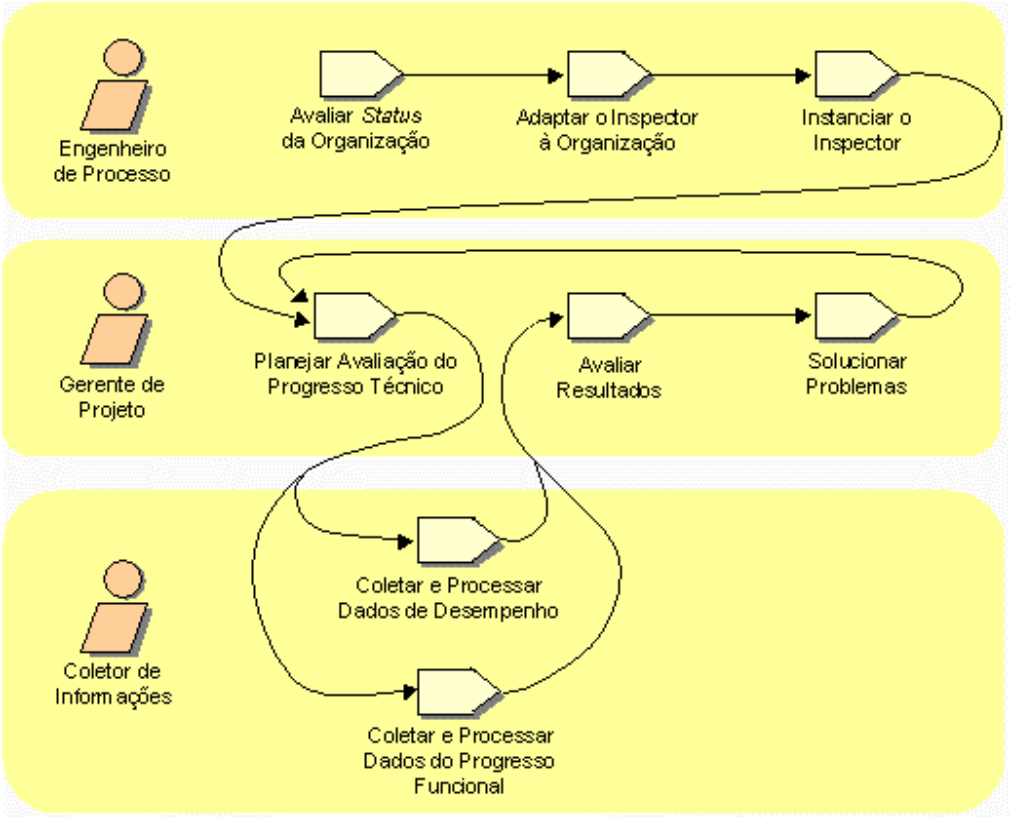


Figura 5.2. Fluxo de atividades do Inspector

As subseções seguintes descrevem brevemente cada uma das atividades definidas no Inspector. A versão estendida do processo [37] contém, em maiores detalhes, todas as informações relacionadas a cada atividade.

5.5.1 Avaliar Status das Métricas na Organização

Para que o Inspector seja implantado com sucesso na organização e instanciado para um projeto específico, é necessário entender o contexto de desenvolvimento do projeto, ou seja, o estado corrente da organização, em relação ao desenvolvimento de software e à utilização de métricas. O Engenheiro de Processo é responsável por capturar este entendimento, a partir dos membros da organização, dos processos de desenvolvimento e gerenciamento aplicados, das ferramentas de suporte utilizadas e da cultura da organização.

É importante identificar áreas onde existem problemas e áreas de possíveis melhorias, visando inserir dentro da organização, uma cultura de armazenamento,

recuperação e análise das métricas de progresso. Como resultado dessa avaliação do *status* da organização é produzido o artefato Visão Geral das Métricas da Organização, que fornece: a situação atual da organização em relação ao desenvolvimento e à utilização de métricas, o nível de conhecimento e habilidade do pessoal da organização, os aspectos positivos e negativos do processo em relação a utilização de métricas, além de indicar as ferramentas empregadas e os artefatos produzidos durante o desenvolvimento.

Tais resultados servirão como base para a implantação do Inspector na organização, bem como configurar a métrica de progresso funcional $\mu_{sistema}$, para os artefatos produzidos pela organização. A avaliação do estado da organização permite ao Engenheiro de Processo:

1. Usar o estado corrente da organização como entrada para implantação do Inspector.
2. Identificar áreas que precisam ser melhoradas. É importante analisar se o processo atual preocupa-se com o acompanhamento e monitoramento de projetos, e onde ele deverá ser adaptado para adquirir essas características.
3. Representar um formalismo, onde se pode explicar aos clientes os problemas atuais, e porque é necessário a inserção de novas técnicas que visam a utilização sistemática de métricas de progresso.
4. Criar uma motivação e um entendimento comum entre os membros da organização que serão diretamente ou indiretamente afetados pelas mudanças que serão introduzidas.

A realização dessa atividade não é uma tarefa trivial, exige um acompanhamento diário do processo de desenvolvimento utilizado pela organização, observação das equipes de desenvolvimento e do ambiente de trabalho [49]. Nessa atividade, é importante utilizar como fonte de entrada, quando possível, algum artefato que represente textualmente, ou via *browser*, o processo de desenvolvimento utilizado na organização, além do Glossário da Organização, para entender os conceitos dos negócios que a organização está envolvida. Abaixo seguem os passos envolvidos na realização dessa atividade.

Coletar informações sobre a organização

É importante gastar uma quantidade de tempo razoável, buscando modelar o comportamento da organização, identificar as técnicas e ferramentas empregadas, além de verificar o processo de desenvolvimento utilizado. A partir desse período de observação, é possível localizar os pontos fortes, bem como as deficiências da organização. Tal passo é essencial para verificar os problemas que serão enfrentados na implantação do Inspector.

Envolver as pessoas

É mais produtivo coletar informações em grupos pequenos. Quando for realizada a avaliação de uma determinada atividade, é muito importante envolver as pessoas que estão trabalhando na organização [55]. Essas pessoas já sabem como as coisas funcionam lá dentro, e são as melhores fontes de idéia para melhorias. Além disso, é interessante fazer os membros da organização sentirem que fazem parte de uma equipe, onde suas idéias e pensamentos têm importância. Assim, as equipes de desenvolvimento e todos os membros envolvidos com o desenvolvimento de projetos de software na organização deverão tomar conhecimento da implantação do Inspector, através da realização de palestras explicativas e reuniões de discussão, onde as opiniões destas pessoas sobre o processo de desenvolvimento e o ambiente de trabalho deverão ser capturadas.

Identificar os artefatos produzidos no desenvolvimento

Produz uma lista com todos os artefatos que podem ser produzidos pela organização, durante o desenvolvimento de um determinado produto. Consiste nos artefatos definidos no processo de desenvolvimento corrente. Além disso, é importante observar características inerentes ao processo de desenvolvimento da organização: paradigma de desenvolvimento (OO, estruturado, etc.), linguagem de modelagem usada (UML, OMT, etc.), linguagem de programação, entre outros.

Verificar o *status* da utilização de métricas

Identifica se existe alguma prática adotada pela organização para captura e análise das métricas de avaliação de progresso, qualidade, tamanho, etc. Observa as deficiências nestas práticas e nas métricas propriamente ditas, verificando a eficácia e precisão dos dados coletados. Verifica os pontos do processo de coleta e avaliação, atualmente utilizados, que devem ser mantidos, quais devem ser atualizados, e quais aspectos do Inspector o processo atual não cobre. Outra informação relevante é verificar se existe um histórico dos projetos anteriores, onde estão armazenados os resultados das diversas métricas utilizadas e as avaliações do progresso destes projetos.

Documentar conclusões

Neste momento, devem ser focalizados os aspectos mais relevantes para a introdução do Inspector. Nesse passo é produzido o artefato Documento de Visão Geral das Métricas na Organização, que documenta a visão geral dos negócios da organização e do processo de desenvolvimento atual, indicando as falhas e apresentando as principais dificuldades a serem ultrapassadas. Além disso, ele deve focalizar as métricas que são utilizadas na organização e como o desenvolvimento é atualmente monitorado, mostrando as contradições com o processo a ser implantado e as deficiências das métricas atuais.

5.5.2 Adaptar o Inspector à Organização

Esta atividade tem como responsável o Engenheiro de Processo, e tem como objetivo, inserir uma cultura de utilização das métricas necessárias para implantação do Inspector dentro da organização. Para tanto é preciso analisar os principais problemas identificados na atividade Avaliar Status das Métricas na Organização, pois, a partir dessa análise, é possível definir o que deverá ser adaptado na organização, e qual a melhor estratégia para se fazer isso. Os passos necessários para se adaptar a organização são:

Identificar as dificuldades

Neste momento, o Engenheiro de Processo deve localizar os principais fatores que podem implicar em problemas para implantação do Inspector. É importante ler o artefato Visão Geral das Métricas na Organização, identificando se os membros da organização já possuem alguma experiência no uso de métricas de avaliação de progresso, se existem restrições de tempo, de paradigma ou de maturidade, que podem implicar numa má aplicação do processo.

Divulgar os benefícios do processo

Geralmente, um dos principais obstáculos na implantação de um novo processo é a resistência de alguns membros da equipe a novas idéias. Muitas vezes, membros da equipe têm medo que os dados sejam usados contra eles, que se levará muito tempo para coletar e avaliar os dados, ou que a equipe irá desviar a atenção visando coletar os dados, ao invés de se preocupar com a construção do sistema [39].

Um processo, independente de sua finalidade (avaliação de progresso, desenvolvimento de sistema, etc.), não é implantado com sucesso, sem a participação e colaboração efetiva de todos os membros da equipe, que devem estar engajados em sua implantação [49]. Desse modo, é importante conscientizar a equipe da necessidade de se utilizar métricas, como os dados serão usados, e mostrar claramente as vantagens de se monitorar o desenvolvimento de sistema. Deve-se tornar claro que as métricas coletadas nunca serão utilizadas para punir ou repreender, individualmente, membros da equipe. Ao contrário, deve-se mostrar que a intenção é identificar problemas antes que eles atinjam proporções maiores, evitando assim, que eles tenham que gastar grande parte do seu tempo realizando manutenções e corrigindo problemas urgentes.

Classificar os artefatos em níveis de privacidade

É importante classificar, em níveis de privacidade, os artefatos produzidos no processo de desenvolvimento da organização, identificados na atividade Avaliar Status das Métricas na Organização. Respeitar a privacidade dos artefatos é essencial para evitar que pessoas não autorizadas tenham acesso a informações sigilosas e, até mesmo,

para evitar que se realize abuso de autoridade. Cada item de dado pode ser classificado em três níveis de privacidade, de acordo com Wiegers [55]:

- **Individual:** somente o indivíduo que produziu o artefato coleta as informações e sabe o que está contido nele. Apesar disso, ele pode ser provido com dados globais do projeto, resultantes do cálculo das métricas dos demais indivíduos do projeto.
- **Equipe do projeto:** o dado é privado aos membros da equipe do projeto. Apesar disso, tais membros podem ter acesso a dados globais da organização que permitirão uma visão organizacional na realização de projetos.
- **Organização:** as informações são compartilhadas entre todos os membros da organização.

Capacitar Pessoas

As maiores dificuldades apresentadas pelos membros das equipes e gerentes de projeto, para coleta e avaliação das métricas de progresso, devem ser identificadas. A partir disso, deve-se definir uma série de cursos rápidos e apresentações que visam extinguir essas dificuldades, aumentar a capacidade de desenvolvimento e estimular o uso de métricas pelo pessoal da organização. Além disso, é importante realizar um projeto piloto que sirva como forma de aproximação dos membros da organização com o Inspector.

Tornar a utilização das métricas de progresso um hábito

Não é necessário consumir tempo para realizar um acompanhamento próximo do projeto. A maioria das atividades necessárias para realização do processo de avaliação pode ser automatizada através de ferramentas e, além disso, representam mais um hábito do que um aborrecimento. A utilização dos formulários e modelos definidos para capturar e expressar os dados (veja Apêndice A), reduzem significativamente o *overhead* para coletar e reportar os dados. Wiegers [55], define algumas dicas importantes para introduzir o uso de métricas na organização:

- **Começar pequeno:** Adquirir uma cultura e infraestrutura de medição leva tempo. Uma vez que o grupo tenha absorvido a idéia de utilização de métricas e tenha se estabilizado, pode-se introduzir as métricas propostas pelo Inspector.
- **Explicar porque:** As equipes devem entender a importância do processo e as motivações que levaram a organização a utilizá-lo.
- **Compartilhar os dados:** As equipes se sentirão muito mais motivadas se forem informadas com resumos e tendências, resultantes da aplicação do processo.
- **Observar as dificuldades:** Verificar se as equipes estão adquirindo a maturidade organizacional desejada, e quais as dificuldades e deficiências que ainda persistem.

5.5.3 Instanciar o Inspector

Esta atividade tem como objetivo adequar o Inspector a um determinado projeto de acordo com as peculiaridades do mesmo. Muitos projetos não produzem determinados artefatos, ou produzem artefatos distintos, de acordo com o seu porte, tipo da aplicação a ser desenvolvida, entre outros fatores. Desse modo é importante entender o projeto e identificar quais artefatos serão realmente produzidos no seu desenvolvimento. Definidos os artefatos, o Inspector deve ser configurado e, consequentemente, suas métricas associadas também, para que ele esteja adequado ao projeto específico. Os passos necessários para se realizar essa atividade são:

Verificar a situação da organização

Esse passo consiste em verificar se a atividade Adaptar o Inspector à Organização foi realizada com sucesso, ou seja, observar se os membros das diversas equipe de desenvolvimento da organização, já estão bem relacionados com a utilização das métricas do Inspector. Caso eles ainda enfrentem alguma dificuldade, é importante eliminá-las nesse momento.

Entender o projeto

É importante fazer uma análise das principais características do projeto para decidir como elas afetam o processo de avaliação de progresso. Identifica o escopo do projeto a

ser desenvolvido e o comportamento do mesmo, a fim de se definir os artefatos necessários e observar o tamanho do projeto para identificar o esforço para o desenvolvimento. Além disso, entender o projeto que será desenvolvido, permite classificá-lo de acordo com seu tipo de aplicação, de tal forma que os seus dados possam ser armazenados e posteriormente recuperados, servindo como uma base histórica, onde comparações entre um projeto em desenvolvimento e experiências similares podem ser realizadas com facilidade.

Definir as etapas e artefatos do projeto

Nesse momento, o Engenheiro de Processo faz uma configuração inicial de $\mu_{sistema}$, definindo o conjunto $ETAPAS$, que indica quais as etapas necessárias para o desenvolvimento de um caso de uso, e os conjuntos $A_j \mid j \in ETAPAS$, indicando quais artefatos deverão ser produzidos ou atualizados em cada etapa do desenvolvimento do caso de uso. Esses conjuntos deverão ser definidos, de acordo com o processo de desenvolvimento utilizado, o tipo de aplicação a ser desenvolvida e o porte da aplicação. Após a definição desses conjuntos é possível instanciar o Modelo de Coleta de Informação sobre o Progresso do Caso de Uso, para que o Coletor possa identificar se o artefato já foi concluído para realização do caso de uso, não foi concluído, ou não é obrigatório para o caso de uso em questão.

Definir unidades de medida e informações para conversão dos dados

Definir as unidades de medida e a precisão que deverão ser utilizadas no projeto, para representação dos valores das métricas e dos gráficos de análise, informando a definição para os coletores. Além disso, é interessante documentar possíveis conversões entre unidades de medidas, de forma a garantir que dados representados em um tipo de unidade diferente possam ser manipulados sem nenhum problema.

Definir mecanismos de acesso às informações

O Engenheiro de Processo deve definir a forma de acesso aos dados, ou seja, como os vários artefatos que deverão ser inspecionados no Inspector poderão ser recuperados para uma posterior avaliação. Esse passo resulta em um artefato denominado

Documento de Acesso aos Dados, que faz um mapeamento dos tipos de dados que deverão ser capturados para seus respectivos mecanismos de acesso. A versão estendida do Inspector [37] apresenta um conjunto de mecanismos de acesso primários que podem ser utilizados.

Modificar o Inspector

Muitas vezes, se torna interessante para um projeto específico, utilizar outras métricas para gerenciamento do projeto, observando características de qualidade, tamanho, entre outras. Nesse passo, o Engenheiro de Processo pode modificar o Inspector, criando um documento contendo as novas características inseridas no processo de avaliação, e retirando características que não serão usadas durante o desenvolvimento desse projeto.

5.5.4 Planejar Avaliação do Progresso Técnico

Essa atividade é realizada de acordo com a demanda, ou seja, quando surgir a necessidade de se avaliar o progresso do projeto, tanto para fornecer uma satisfação ao cliente, quanto como forma do Gerente de Projeto acompanhar o desenvolvimento. Tem como responsável o Gerente de Projeto, que deve produzir o artefato Plano de Avaliação do Progresso Técnico, definindo o escopo da avaliação, quando ela será realizada, quem é o responsável pela coleta das métricas e quais os resultados esperados. Sua realização consiste em três passos:

Determinar o escopo e a data de avaliação

O Gerente de Projeto deve definir qual o escopo da avaliação de progresso, e quando os dados deverão ser avaliados. O escopo indicará quais casos de uso serão considerados na avaliação de progresso, e qual o intervalo de tempo essa avaliação irá cobrir.

Determinar o membro de cada equipe responsável pelas métricas

Fazer um mapeamento indicando, para cada equipe, um responsável pela coleta das visões de desempenho e funcionalidade. Consiste em uma tabela simples que define um

Coletor para cada equipe, indicando os casos de uso que ele deverá coletar as informações.

Determinar as metas da avaliação

Determinar quais os resultados esperados para as métricas que serão coletadas e o *status* do projeto como um todo. O valor esperado pode ser estimado através de relatos de experiências com projetos semelhantes, ou ser apenas um incremento percentual esperado pelo gerente como resultado da avaliação. A definição dessas metas durante o planejamento é importante, pois representa um parâmetro de comparação, que indicará se os resultados obtidos correspondem ao esperado ou representam alguma deficiência.

5.5.5 Coletar e Processar Dados de Desempenho

Essa atividade visa recuperar as informações que serão utilizadas para análise do desempenho do projeto, ou seja, os gráficos Gantt e PERT, e as métricas de desempenho $\mu_{concluído}$, $\mu_{atraso\ médio}$ e $\mu_{novas\ ativdades}$. Cada Coletor, definido na atividade Planejar Avaliação do Progresso Técnico, deve realizar essa atividade, coletando as informações referentes ao escopo que lhe foi atribuído no Plano de Avaliação do Progresso Técnico. Os principais passos para coleta e cálculo das métricas de desempenho são:

Acessar dados de desempenho da equipe

Nesse passo, cada Coletor recupera as informações de desempenho da equipe de desenvolvimento que ele é responsável, ou seja, o gráfico Gantt de atividades e o gráfico PERT atuais. O Coletor deve realizar uma reunião com a equipe, identificando as atividades de desenvolvimento realizadas no escopo da avaliação. Ele deve verificar para cada atividade, a data de início e fim, e caso ela não tenha sido concluída, realizar uma estimativa para indicar o quanto já se foi concluído. Essa estimativa fica a cargo do coletor, que pode utilizar alguma técnica mais formal como análise de pontos de função [10], ou basear-se na sua experiência e na conversa com o responsável pela atividade.

Calcular o quanto a equipe concluiu das atividades planejadas

Nesse passo o Coletor de Informações deverá calcular o valor da métrica $\mu_{concluído}$ da equipe que ele é responsável. O cálculo baseia-se na avaliação dos gráficos Gantt estimados e atuais, coletados no passo anterior, e já foi apresentado com mais detalhes na Seção 5.2.1, que apresenta as métricas inseridas na visão de desempenho.

Calcular a porcentagem média de atraso da equipe

Nesse passo o Coletor de Informações deverá calcular o valor da métrica $\mu_{atraso\ médio}$ da equipe que ele é responsável. O cálculo baseia-se na avaliação dos gráficos Gantt estimados e atuais, identificando a porcentagem média de atraso por atividade. A forma de aquisição de $\mu_{atraso\ médio}$ foi apresentada com detalhes na Seção 5.2.1.

Calcular a porcentagem de novas atividades que surgiram

Aqui, o Coletor deverá calcular o valor da métrica $\mu_{novas\ atividades}$ da equipe que ele é responsável. Essa métrica visa avaliar quão bom está sendo o planejamento, observando se atividades não esperadas estão surgindo, prejudicando o desempenho da equipe. A forma de aquisição de $\mu_{novas\ atividades}$ foi apresentada com detalhes na Seção 5.2.1.

5.5.6 Coletar e Processar Dados do Progresso Funcional

Nessa atividade, cada Coletor deve capturar as informações relacionadas ao progresso dos casos de uso que fazem parte do seu escopo. O conjunto de casos de uso que o Coletor ficará responsável pela coleta de informações é definido na atividade Planejar Avaliação do Progresso Técnico. A saída dessa atividade é, para cada caso de uso, o Modelo de Coleta de Informação sobre o Caso de Uso preenchido, com o progresso funcional do caso de uso calculado. Essa atividade é dividida nos seguintes passos:

Buscar as informações gerais sobre o projeto

Corresponde ao preenchimento inicial do Modelo de Coleta de Informação de Progresso sobre o Caso de Uso. Inicia-se documentando os dados gerais de identificação

do caso de uso, ou seja, seu nome, número de identificação, uma breve descrição de sua finalidade, o nome do projeto ao qual ele pertence, a equipe responsável, etc. Além disso, as informações sobre a coleta que está sendo realizada (data da coleta, responsável, etc.), e os valores obtidos na avaliação anterior, também devem ser documentados no modelo.

Definir a prioridade de cada caso de uso

Os casos de uso devem ser priorizados, de forma a garantir que a conclusão de casos de uso mais complexos e que necessitem de mais tempo para serem realizados, indiquem um maior progresso para o projeto. Os indicadores citados na Seção 4.4.1, representam um importante auxílio, mostrando fatores que podem ser considerados durante a definição dos pesos dos casos de uso. Além disso, o responsável pelo caso de uso deverá ser consultado, pois ele é a pessoa que pode fornecer informações mais concretas sobre a prioridade do caso de uso. Como resultado, todos os casos de uso deverão estar priorizados, configurando o ambiente para posterior cálculo de $\mu_{sistema}$.

Configurar as etapas necessárias para cada caso de uso

O Coletor de Informações deve, para cada caso de uso que faz parte do seu escopo de coleta, identificar quais são as etapas necessárias para o seu desenvolvimento. Desse modo, esse passo configura o conjunto ETAPAS para um determinado caso de uso. O valor padrão para esse conjunto foi apresentado na Seção 4.2.1.

Configurar os artefatos necessários para cada caso de uso

Inicialmente, na atividade Instanciar o Inspector, o Engenheiro de Processo definiu todos os artefatos que poderão ser produzidos durante o desenvolvimento do projeto, e seus respectivos critérios de inspeção. Aqui, o Coletor de Informações verifica, para cada caso de uso pertencente ao seu escopo, quais artefatos o caso de uso específico realmente irá produzir. Desse modo, esse passo configura, para cada caso de uso, os conjuntos $A_j \mid j \in ETAPAS$, correspondendo aos conjuntos que indicam os artefatos que devem ser produzidos em cada etapa do caso de uso.

Coletar as informações de progresso de cada caso de uso

Esse passo apresenta uma complexidade relativamente grande, pois o Coletor de Informações deverá buscar as informações relativas ao progresso funcional de cada caso de uso. Inicialmente, o Coletor de Informações verifica quais etapas foram indicadas como necessárias para o desenvolvimento do caso de uso, e quais artefatos devem ser produzidos em cada etapa. As etapas e artefatos indicados como não necessários, deverão ter a propriedade "Não Necessário" assinalada no Modelo de Coleta de Informação sobre o Progresso do Caso de Uso.

A partir disso, é feito um rastreamento dos artefatos necessários, de acordo com o mecanismo de acesso aos dados definido pelo Engenheiro de Processo. O Coletor de Informações deve inspecionar se tais artefatos contêm as informações relacionadas ao desenvolvimento do caso de uso e à incorporação de suas funcionalidades no sistema. A inspeção dos artefatos deve obedecer aos critérios definidos no artefato Critérios de Avaliação dos Artefatos. O resultado desse passo é o Modelo de Coleta de Informação sobre o Progresso do Caso de Uso preenchido para cada caso de uso, indicando os artefatos que foram avaliados positivamente, negativamente ou não são necessários para realização do caso de uso.

Calcular o progresso funcional de cada caso de uso

Esse passo é executado quase que simultaneamente ao passo anterior, de modo que, ao mesmo tempo em que os artefatos de uma etapa são inspecionados, já se pode calcular o progresso obtido nesta etapa. No final, basta apenas calcular o progresso funcional do caso de uso que é derivado a partir dos resultados obtidos na inspeção das etapas. O cálculo necessário para se encontrar o valor de $\mu_{caso\ de\ uso}$ foi apresentado na Seção 4.2.1.

5.5.7 Avaliar Resultados

Consiste na atividade mais complexa, e que exige maior esforço durante a aplicação do Inspector. Nesse momento, o Gerente de Projeto deve obter uma visão geral do

progresso do projeto, que corresponde às duas visões de progresso, mostradas em detalhe na Seção 5.2.

Ele deve analisar os dados fornecidos pelos coletores, fazendo uma comparação do que foi realizado com o que fora inicialmente planejado. Além disso, ele deve observar mudanças substanciais nos gráficos PERT das equipes, monitorar o desempenho geral das equipes de desenvolvimento ($\mu_{concluído}$, $\mu_{atraso\ médio}$ e $\mu_{novas\ atividades}$), calcular o progresso funcional do projeto ($\mu_{sistema}$) e, a partir desse cálculo, fazer ponderações sobre os resultados obtidos, através da representação dos dados em gráficos e tabelas, observando incrementos em relação à última avaliação, e analisando tendências para o futuro. Os principais passos para avaliação dos resultados são:

Verificar dados entregues

Após a recuperação dos dados é importante fazer uma verificação, observando a veracidade dos dados entregues pelos coletores, e se eles estão representados corretamente. O Gerente de Projeto deve responder uma lista de verificação, que contém um conjunto de questões, que verificam se os dados coletados contêm as propriedades desejadas. A Tabela 5.3 consiste em um exemplo de lista de verificação, retirada a partir do PSM (Pratical Software Measurement), que define boas práticas e técnicas para uso de métricas [30].

Dados correntes	Os diagramas recebidos estão sendo entregues na data prevista?
Atributos dos dados	Os gráficos e dados são consistentes com o recomendado?
Unidades de medida	As mesmas unidades de medida estão sendo utilizadas por todas as equipes? Elas estão de acordo com as unidades definidas na atividade Instanciar o Inspector?
Conteúdo dos dados	Os valores representados são aceitáveis?
Dados completos	Todos os dados necessários para descrição das visões foram apresentados? Os dados apresentados são realmente necessários?

Tabela 5.3. Exemplo de lista de verificação dos dados

Normatizar dados capturados

Antes dos dados serem analisados, os gráficos e dados que apresentarem alguma deficiência ou divergência dos padrões devem ser normatizados, para permitir futuras comparações. A normatização de um dado deve ser realizada cuidadosamente, seguindo as regras de conversão de medidas ou escala, documentadas em Instanciar o Inspector.

Além disso, é preciso manter a consistência entre o gráfico da equipe e o gráfico normatizado.

Comparar Gantt de atividades atuais com estimados

Uma das mais tradicionais formas de se avaliar o desempenho do projeto consiste no monitoramento dos gráficos Gantt, observando se as atividades inicialmente planejadas foram concluídas ou não, identificando atividades em atrasos por parte de uma determinada equipe. O Gerente de Projeto pode agendar uma reunião com cada equipe, visando identificar mais precisamente os problemas que estão sendo enfrentados.

Uma melhor forma de visualizar o atraso das atividades é realizar uma sobreposição dos gráficos atual e estimado, resultando em um terceiro gráfico Gantt, como mostrado na Figura 5.3. Os atrasos encontrados podem ser classificados em aceitáveis ou inaceitáveis, indicando atividades que apresentam os maiores problemas.

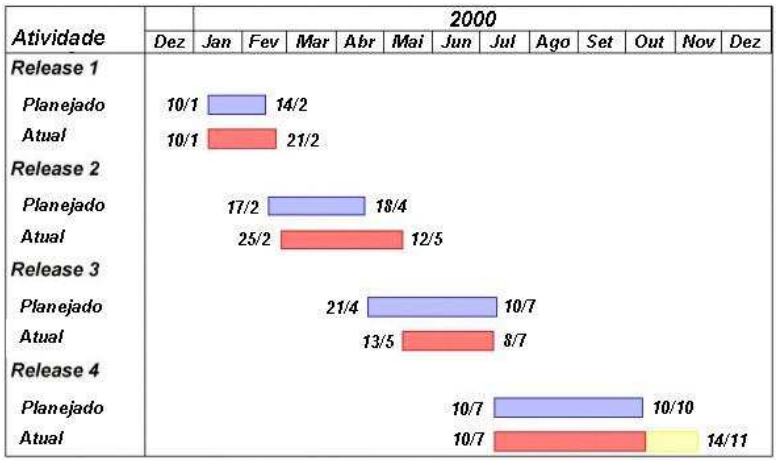


Figura 5.3. Exemplo de gráfico Gantt atual x estimado

Comparar o caminho crítico de atividades atual com anterior

Consiste em comparar, para cada equipe, o gráfico PERT atual, entregue pelo coletor, com o gráfico PERT armazenado na avaliação anterior. Deve ser verificado se o caminho crítico ainda é o mesmo ou se foi alterado. Mudanças no caminho crítico indicam que houve algum problema entre a avaliação anterior e a atual, ou seja, alguma atividade crítica atrasou. Nesse caso, é importante verificar a atividade que atrasou e

conversar com o responsável para identificar o que aconteceu. O Gerente de Projeto deve informar os membros da equipe sobre as atividades que pertencem ao caminho crítico atual, conscientizando a importância dessas atividades, e a necessidade de se tentar evitar atrasos na realização das mesmas.

Analisar as métricas de desempenho de cada equipe

Corresponde à análise das métricas de desempenho definidas pelo Inspector ($\mu_{concluído}$, $\mu_{atraso\ médio}$ e $\mu_{novas\ atividades}$). Cada uma das métricas focaliza diferentes aspectos de desempenho, sendo que, a avaliação delas permite obter uma visão precisa do desempenho de cada equipe.

O valor da métrica $\mu_{concluído}$, que indica o quanto das atividades planejadas foram realizadas, deve ser o mais próximo possível de 1 (100%), pois valores baixos indicam que a equipe não está conseguindo realizar as atividades como deveriam. O valor de $\mu_{atraso\ médio}$, mostra o quanto, proporcionalmente, a equipe está atrasando por atividade inicialmente planejada. O valor obtido por essa métrica deve estar próximo de 0 (0%), valores altos indicam que as atividades não estão sendo realizadas no tempo previsto. Por fim, a métrica $\mu_{novas\ atividades}$ indica, quantitativamente, o surgimento de atividades que não estavam planejadas para a equipe. O valor dessa métrica deve ser próximo de 0 (0%), valores altos representam problemas no planejamento, que afetam negativamente o desempenho.

O Gerente de Projeto também pode representar os valores obtidos através de gráficos de linha, indicando a variação do desempenho da equipe durante o desenvolvimento do projeto. Observando, dessa forma, se resultados negativos de avaliações anteriores já foram superados, e quais equipes tiveram maior decréscimo na produtividade.

Calcular o progresso funcional do projeto

Após todos os coletores entregarem o progresso dos casos de uso que ficaram sob suas responsabilidades, é possível calcular o progresso funcional de todo o sistema

($\mu_{sistema}$), e de cada etapa necessária para seu desenvolvimento. Nesse momento, o Gerente de Projeto deve preencher o Modelo de Coleta de Informação sobre o Progresso do Projeto, indicando os casos de uso do projeto, com suas respectivas prioridades e progresso funcional. Em seguida é calculado o progresso funcional do sistema, de acordo com as regras e fórmulas definidas na Seção 4.2.

Criar tabela resumo do progresso funcional

Os dados obtidos durante o cálculo do progresso funcional deverão ser representados sobre a forma de tabela, facilitando o entendimento dos resultados alcançados. A tabela resumo, já detalhada e exemplificada (Tabela 4.2), representa a maneira mais fácil de se analisar o *status* atual da visão de funcionalidade, identificando claramente resultados problemáticos na realização de um caso de uso. Além da tabela resumo, é interessante representar também uma tabela indicando o incremento percentual no progresso funcional de cada caso de uso, etapa por etapa. Casos de uso que apresentarem incrementos percentuais inferiores aos esperados, indicam a existência de problemas durante seu desenvolvimento, devendo ser esclarecido, junto ao responsável pelo caso de uso, as dificuldades que vêm sendo encontradas.

Criar gráficos para visualizar os incrementos

Nesse passo, devem ser gerados gráficos de linha indicando a variação de progresso nas funcionalidades do sistema e dos casos de uso que o compõem. Gráficos são uma excelente maneira de se observar a variação do progresso em relação ao tempo de desenvolvimento, mostrando visualmente, progressos abaixo do esperado, e a tendência à atraso na realização de determinados casos de uso.

Analisar tendências

Devem ser observadas tendências nos gráficos gerados sobre o progresso funcional dos casos uso e do sistema, de modo a identificar atrasos que podem surgir de pequenos problemas acumulados, muitas vezes desapercebidos durante o desenvolvimento, observando também a necessidade de se adaptar o cronograma. Uma técnica simples para observar tendência é, a partir do gráfico de linha gerado no passo anterior, traçar

uma linha reta que corresponda a possível continuidade no progresso funcional do caso de uso ou sistema. A partir disso, é possível verificar a existência de um possível atraso, que pode ser aceitável ou precisar de cuidado especial, dependendo de sua magnitude.

Agrupar as duas visões observadas

O Gerente de Projeto deve manter a consistência entre as duas visões de progresso obtidas, para tanto, é interessante realizar uma sobreposição destas visões, relacionando os casos de uso problemáticos com as atividades que foram definidas para construção do caso de uso. Deve-se verificar se a duração estimada para realização da atividade condiz com a duração prevista para o desenvolvimento do caso de uso, segundo a análise de tendências previamente realizada. Caso haja problemas de consistência, estes deverão ser documentados, e o pessoal envolvido deverá ser chamado para esclarecer a questão.

Documentar conclusões

Consiste em documentar, no artefato Avaliação do Progresso Técnico, todos os dados que foram analisados ou gerados durante a realização dessa atividade, ou seja, os gráficos Gantt atuais e estimados, gráfico PERT atualizado, métricas de desempenho, tabela resumo do progresso funcional, tabela de incremento funcional, gráficos de linha do progresso do sistema e dos casos de uso, análises de tendências e todas as conclusões que foram tiradas durante a avaliação do progresso técnico. Os problemas encontrados devem ser documentados em uma tabela simples, que serve como fonte de entrada para a atividade seguinte, onde os problemas deverão ser solucionados.

5.5.8 Solucionar Problemas

Nessa atividade, o Gerente de Projeto busca soluções para os problemas encontrados durante a avaliação do *status* do projeto, em termos do seu progresso técnico. Cada problema deverá ser trabalhado, visando sua eliminação. O Gerente de Projeto, juntamente com as pessoas envolvidas deverão encontrar as possíveis soluções para cada problema, e identificar a solução mais adequada à situação atual do projeto. Os passos a seguir, consistem em um guia rápido para tomada de decisões, e devem ser realizados para cada problema identificado no artefato Avaliação do Progresso Técnico:

Detalhar o problema

O primeiro passo para a solução de um problema é buscar informações sobre o mesmo, para que se possa identificar a sua profundidade. Desse modo, é importante identificar o problema na Avaliação do Progresso Técnico, e descrever em detalhes a existência de dificuldades na execução de alguma atividade ou realização de algum caso de uso. Muitas vezes, projetos não foram planejados de acordo com a realidade, dificuldades surgem devido a necessidade da utilização de uma nova tecnologia, pessoal alocado para a realização de uma atividade não foi corretamente escolhido, fatores externos atrapalham a execução da atividade, entre outros fatores, que nesse momento, devem ser identificados.

Encontrar possíveis soluções

Após a descrição detalhada do problema, o Gerente de Projeto deve realizar um estudo para encontrar as possíveis soluções para tais problemas. Nesse momento, é muito importante envolver os *stakeholders* que apresentam bom conhecimento técnico e do domínio do problema, pois eles possuem uma visão completa do problema, e poderão ajudar a encontrar soluções [33]. As soluções encontradas devem ser documentadas e apresentadas no Documento de Solução dos Problemas Identificados, que deve conter as informações relativas ao problema que a solução está relacionada, a descrição dos passos para execução da solução e os riscos que são inerentes àquela solução.

Escolher a solução adequada

O Gerente de Projeto deve selecionar a solução que será executada, com o intuito de resolver, ou pelo menos minimizar, o problema observado. Após encontrar as possíveis soluções no passo anterior, o gerente observará as vantagens e desvantagens de se aplicar cada uma. Humphrey [27], cita a facilidade de implantação, a confiabilidade, a efetividade, os riscos inerentes e a cobertura, como sendo importantes fatores que devem ser considerados na escolha de uma determinada solução. Uma tabela, contendo esses e outros fatores relevantes, pode ser montada para facilitar a escolha da solução mais adequada, como mostrado na Tabela 5.4.

O resultado do mapeamento indicará quais vantagens cada solução apresenta e, de acordo com as prioridades do projeto e da organização, o Gerente de Projeto poderá escolher a solução mais adequada. A definição dessa tabela representa uma alternativa simples, para se identificar qual a melhor solução de um determinado problema. Gilb [20], apresenta um estudo detalhado de como se avaliar o impacto que a aplicação de uma determinada solução para um problema, implicará para o projeto.

Alternativa de Solução	Facilidade de Implantação	Confiabilidade	Efetividade	Cobertura	Riscos não problemáticos
1	✓	✓			
2			✓	✓	
3	✓	✓		✓	✓
4	✓				✓

Tabela 5.4. Exemplo de tabela de apoio a tomada de decisão

Executar a solução escolhida

Nesse passo, o Gerente de Projeto deve aplicar a solução escolhida seguindo a descrição e as tarefas necessárias, indicadas no Documento de Solução dos Problemas Identificados. Durante a execução da solução, ele deve estar sempre atento aos riscos associados à solução, monitorando o processo de implantação, e observando se os resultados que estão sendo obtidos são os esperados. Caso algum desses riscos ocorra, ele deve ser tratado de acordo com o descrito no Documento de Solução dos Problemas Identificados, de modo a evitar que o mesmo represente prejuízo para o projeto. Além disso, o surgimento de novos riscos também deve ser observado e monitorado.

Revisar e atualizar os dados do projeto

O Gerente de Projeto deve verificar em quais pontos será necessário mudar os artefatos de apoio (Plano de Projeto, Plano de Iteração, etc.), para que eles representem realidade atual do projeto. Mudanças no cronograma, nas equipes de desenvolvimento, nas atividades, e até mesmo nos diagramas e outros artefatos do projeto, podem ter acontecido, de forma a solucionar um determinado problema encontrado.

5.6 Limitações do Processo

O Inspector representa a proposta de um processo sistemático de acompanhamento de projetos, que visa monitorar o progresso do projeto, como forma de se controlar o desenvolvimento do sistema. Apesar do Inspector tentar ser o mais completo possível, algumas limitações podem ser observadas, mas estas não correspondem a problemas graves que impossibilitem o uso eficaz do processo.

Uma limitação observada é que o Inspector considera somente o progresso no desenvolvimento, desconsiderando aspectos não funcionais, como por exemplo, a qualidade do sistema que está sendo produzido, aspectos organizacionais ou cognitivos, que influenciam o desenvolvimento, entre outros.

Outra limnitação do Inspector é o fato dele não considerar individualmente cada membro da organização, nem definir métricas para rastreamento do seu nível de habilidade, conhecimento, e o quanto isso influencia no progresso de um determinado projeto. O que o Inspector procura fazer é fornecer informação relacionada a produtividade de cada equipe de desenvolvimento pertencente a um determinado projeto, sem observar cada membro da equipe individualmente.

O Inspector, apesar de definir um conjunto relativamente reduzido de métricas para avaliação de progresso, necessita da implementação de uma ferramenta que o automatize, pelo menos parcialmente, permitindo um cálculo automático das métricas definidas, de forma a agilizar o processo. Isso se deve ao fato que, para projetos grandes, pode se tornar inviável manipular a quantidade de informações envolvidas com a captura das duas visões de progresso. Uma breve descrição de uma suposta ferramenta, que tornasse o uso do Inspector mais prático e eficiente, pode ser encontrada na Seção 7.1.4.

5.7 Considerações Finais

O Inspector consiste em uma importante fonte de apoio ao desenvolvimento de projetos de software. Monitorar o progresso eficientemente é o maior objetivo deste

processo, e representa um passo essencial para a aquisição de uma maior maturidade no desenvolvimento de software. Projetos controlados e gerenciados tendem a obter sucesso no desenvolvimento, de forma que, as atividades definidas pelo Inspector visam tornar o pessoal mais capacitado para encontrar problemas no desenvolvimento e buscar soluções, através de uma melhoria de comunicação entre as equipes e seus membros.

A aquisição e análise dos dados de progresso sob dois pontos de vista, representa uma peculiaridade interessante do Inspector, monitorando o progresso do ponto de vista do desempenho das equipes que participam do projeto, e das funcionalidades adquiridas a partir da realização dos casos de uso, ou seja, dos aspectos funcionais que devem estar incorporados no sistema final. Essa divisão de foco, além de representar com maior precisão cada um destes aspectos, também facilita a análise dos dados e, a forma de tratar e representar os mesmos.

Além disso, outra característica importante do Inspector é a descentralização da responsabilidade de monitoramento dos projetos, através da definição de um conjunto de responsáveis para realização do processo, evitando que o trabalho fique todo nas mãos do Gerente de Projeto, como normalmente acontece. Esse processo usa um modelo de comunicação semelhante ao modelo hierárquico tradicional de gerenciamento de projetos, representado na Figura 2.3, onde o Gerente de Projeto tem sua autoridade reconhecida, e praticamente não existe comunicação entre membros de diferentes equipes [16]. Uma diferença do modelo hierárquico tradicional, para o modelo definido pelo Inspector, é a maior mobilidade do Gerente de Projeto, que se comunica com todos os membros das equipes, não somente com o líder da equipe, como definido no modelo tradicional. Além disso, o Inspector permite a definição da privacidade e dos mecanismos de acesso aos dados, de acordo com o projeto a ser desenvolvido, variando, desse modo, a comunicação de projeto para projeto.

Alguns conceitos relativos ao Inspector foram descritos resumidamente neste capítulo. Além disso, nem todas as propriedades da versão estendida do processo, inseridas durante a definição dos responsáveis, artefatos e atividades, foram representadas neste capítulo, que focalizou as características principais do processo. A versão completa do Inspector pode ser encontrada em [37].

Capítulo 6

Aplicação do Inspector a um Projeto Real

Este capítulo mostra os problemas enfrentados e os resultados obtidos durante a aplicação do Inspector a um projeto real. A realização deste estudo de caso buscou identificar possíveis falhas no Inspector, avaliando a facilidade de implantação, de utilização e a efetividade dos resultados obtidos na utilização sistemática do processo e das métricas que ele incorpora. O estudo de caso apresenta também as vantagens de se utilizar o Inspector, sugerindo o processo como um modelo eficaz de apoio ao gerente de projeto para avaliação do progresso técnico do projeto e do desempenho das equipes de desenvolvimento.

Durante o estudo de caso realizou-se o acompanhamento e monitoramento de parte de um projeto, denominado Nota Fiscal Virtual, através do uso sistemático do Inspector. O projeto Nota Fiscal Virtual, realizado pela Empresa Municipal de Informática do Recife (Emprel), visa a construção de um sistema que tem como objetivo principal tornar a relação da Prefeitura da Cidade do Recife com os seus contribuintes mais amigável, eficiente e transparente. A Internet tem sido uma grande aliada na implementação de ações que buscam melhorar essa relação, tornando disponível diversos serviços, fornecendo informações e servindo como um canal de comunicação com o contribuinte. Os principais serviços oferecidos pelo sistema são a emissão da nota fiscal de serviço avulsa (NFSA) e a autorização de impressão de documentos fiscais (AIDF). Este projeto é um embrião para no futuro tornar disponível a emissão das notas fiscais de serviço autorizadas pela Prefeitura da Cidade do Recife, desobrigando o

contribuinte de confeccionar estes documentos tipograficamente e melhorando a relação contribuinte/ fisco. A importância do projeto está focada na melhoria do atendimento ao contribuinte, através da ampliação dos serviços na Internet que além de desobrigar o contribuinte de comparecer fisicamente ao prédio sede da Prefeitura para solicitar e obter os serviços, tem o horário de atendimento ampliado para 24 horas por dia. Para o fisco municipal, o projeto é importante porque torna a relação com o contribuinte mais ágil e democrática.

O Nota Fiscal Virtual é um projeto de médio porte que teve a duração de pouco mais de sete meses, tendo sido iniciado no mês de maio e concluído em dezembro de 2000. O estudo de caso iniciou quando o projeto já estava em andamento e teve a duração de dois meses e quinze dias, tendo sido iniciado no dia 28 de setembro e finalizado no dia 13 de dezembro. Neste período foram realizadas as atividades definidas no Inspector para monitorar o desenvolvimento do projeto, bem como produzidos os artefatos estabelecidos como resultados dessas atividades.

Devido ao fato da organização e o projeto serem reais, houve algumas limitações impostas para evitar que a aplicação do Inspector causasse impacto negativo no esforço necessário para o desenvolvimento, no pessoal alocado ou no cronograma do projeto. A Figura 6.1, apresenta quais atividades do Inspector foram aplicadas completamente, quais foram parcialmente aplicadas e quais não foram aplicadas no projeto Nota Fiscal Virtual.

As limitações na aplicação do Inspector estarão sendo apresentadas com maiores detalhes durante a descrição do trabalho realizado. Apesar destas dificuldades, a Emprel atendeu a maioria dos requisitos básicos para utilização do Inspector, ou seja, apresentou um processo de desenvolvimento orientado a objetos [50], utilizando a UML (*Unified Modeling Language*) [8] como linguagem de modelagem dos dados. Além disso, as limitações enfrentadas não influenciaram decisivamente nos resultados obtidos, apenas limitando a aplicação das soluções encontradas para os problemas identificados e, algumas vezes, dificultando o acesso a algumas informações.

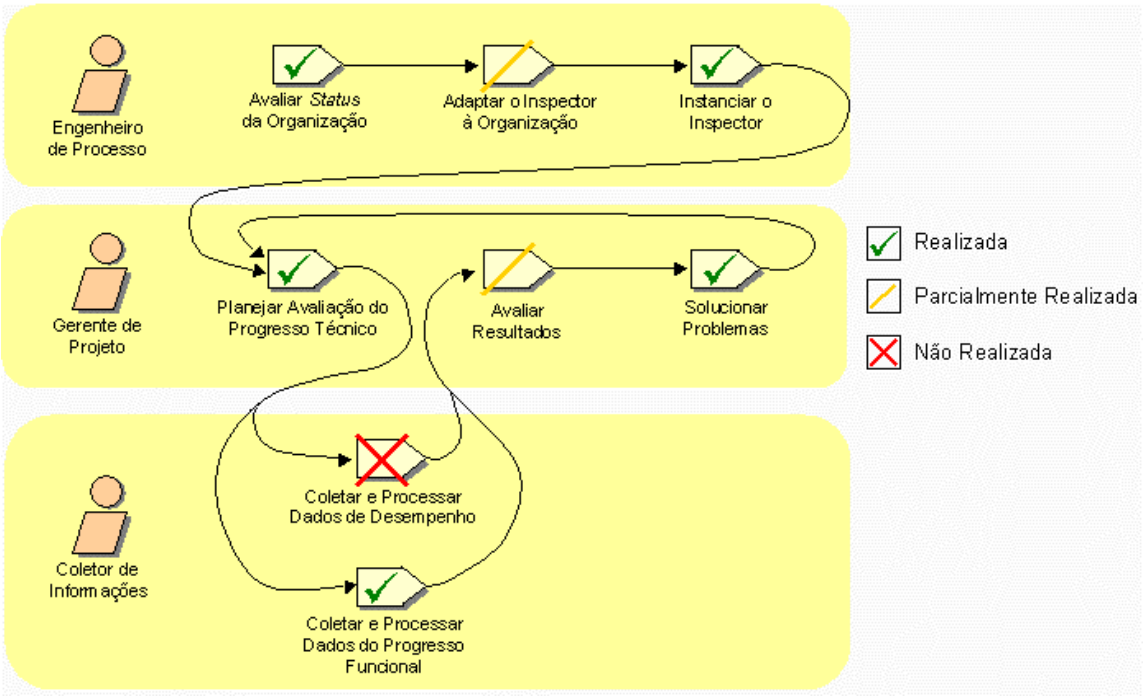


Figura 6.1. Atividades realizadas na aplicação do Inspector

As seções seguintes descrevem com maiores detalhes aplicação do Inspector ao Nota Fiscal Virtual, apresentando as atividades que foram realizadas durante o acompanhamento deste projeto. Além disso, serão apresentados também os problemas enfrentados, as observações consideradas relevantes e os resultados obtidos durante as avaliações de progresso realizadas. A partir da análise deste estudo de caso é apresentada uma série de conclusões e considerações sobre a viabilidade da aplicação do Inspector em organizações que desenvolvem software e as melhorias e adaptações que este estudo de caso resultou para o Inspector.

6.1 Avaliando o Status das Métricas na Organização

A primeira atividade realizada foi Avaliar o Status das Métricas na Organização, seguindo o indicado pelo Inspector, onde se buscou obter uma visão geral da organização quanto ao processo de desenvolvimento e à utilização de métricas para acompanhamento do processo e do produto em desenvolvimento.

Essa atividade teve duração de duas semanas e não apresentou grandes problemas, sendo realizada através de visitas à Emprel, onde o processo de desenvolvimento e o ambiente de trabalho das equipes de desenvolvimento puderam ser observados. Durante a execução dessa atividade tornou-se clara a importância de se envolver as pessoas, ou seja, os membros da organização na recuperação das informações. Na organização em questão, elas se mostraram bastante receptivas às solicitações, havendo sempre um membro da equipe oferecendo total atenção durante a avaliação, retirando dúvidas sobre o processo de desenvolvimento e as relações entre os membros das equipes de desenvolvimento. Outra importante fonte de informação foi a *homepage* da organização e do projeto, que ofereceram informações significativas sobre o processo de desenvolvimento atual.

O resultado produzido pela execução dessa atividade foi o artefato Visão Geral das Métricas na Organização, que descreveu as principais características do processo de desenvolvimento, os problemas encontrados nesse processo, os artefatos que poderão ser produzidos durante o desenvolvimento de um projeto na organização, bem como o *status* da utilização de métricas. A versão completa deste artefato pode ser encontrada em [37].

Resumidamente, observou-se que a organização encontrava-se em um estágio de mudança no processo de desenvolvimento, deixando o paradigma estruturado e a produção não controlada, baseada na experiência dos gerentes, e partindo para inserção de uma nova metodologia baseada no RUP [17], contendo atividades de desenvolvimento voltadas para o paradigma orientado a objetos.

O ambiente de desenvolvimento da organização utiliza o Rational Rose [46] como ferramenta de modelagem (linguagem de modelagem UML), IBM Visual Age for Java [40] como ferramenta de implementação (linguagem Java), Microsoft Project98 [45] como ferramenta para gerenciamento de cronogramas (ainda não usado na prática) e não possui ferramentas de teste.

Observou-se também que a organização não apresenta preocupação em monitorar o desenvolvimento a partir de métricas bem definidas. O acompanhamento de projetos

fica mais por conta da experiência do gerente de projeto e da comunicação de problemas surgidos no desenvolvimento. Não existe a tentativa de se verificar a existência de um problema antes que ele atinja proporções maiores. Além disso, não foi observada a existência de nenhuma métrica que verifique a qualidade do produto que está sendo produzido.

Ainda no artefato Visão Geral das Métricas na Organização foi documentado que o Nota Fiscal Virtual, além de ser um projeto cujo desenvolvimento já estava previsto, está servindo como um projeto piloto para aprendizado da linguagem de programação Java e utilização da nova metodologia. Nele foram construídos a maioria dos artefatos definidos no RUP que são relevantes para o Inspector (apresentados na Seção 4.3), exceto os artefatos voltados para construção de um protótipo inicial e voltados para teste, pois neste projeto os testes foram realizados sem nenhum planejamento.

A organização, por estar em um processo de transição de um paradigma estruturado para o paradigma orientado a objetos e implantando uma nova metodologia mais elaborada, sente algumas dificuldades no desenvolvimento. O planejamento e o gerenciamento ainda baseiam-se na experiência do gerente, que determina as datas, mas não utiliza ferramentas para registrá-las e acompanhar se elas estão sendo obedecidas.

Concluiu-se que a implantação do Inspector poderia enfrentar dificuldades de aceitação por parte dos membros das equipes de desenvolvimento, que poderiam não entender sua importância. Assim, para evitar problemas de aceitação, foi importante tornar claro para os membros que a utilização de métricas de progresso não iria prejudicar ninguém individualmente. Além disso, a importância do uso de métricas teve que ser enfatizada. A estratégia utilizada foi disseminar as métricas que seriam coletadas, como elas seriam usadas, quem iria trabalhar com elas e o que mudaria para cada membro com a inserção do processo.

6.2 Adaptando a Organização

Essa atividade pretende capacitar a organização para utilização do Inspector em seus projetos através da inserção de uma cultura de utilização de métricas de progresso.

Ela foi aplicada parcialmente neste estudo de caso, devido às limitações impostas para sua realização, que visou não influenciar no desenvolvimento do projeto Nota Fiscal Virtual. Desse modo, alguns passos desta atividade que necessitavam da participação efetiva da equipe de desenvolvimento e consumiriam tempo da mesma, foram realizados somente com um único membro da equipe de desenvolvimento.

Assim, os passos “Divulgar os benefícios do processo”, “Capacitar pessoas” e “Tornar a utilização das métricas de progresso um hábito” foram realizados sobre um único membro da equipe de desenvolvimento, que obteve um maior entendimento do Inspector e das métricas que ele envolve, observando como os dados são coletados, calculados e avaliados.

Já os passos “Identificar as dificuldades” e “Classificar os artefatos em níveis de prioridade”, por não precisarem envolver toda organização, foram aplicados completamente. O primeiro representa uma síntese dos problemas observados na organização durante a realização da atividade Avaliação do Status das Métricas na Organização. O segundo indicou quais dados seriam visíveis para o coletor de informações (todos os artefatos produzidos para realização dos casos de uso da equipe sob sua responsabilidade), quais dados seriam visíveis para a equipe (desempenho da equipe e progresso dos casos de uso da equipe) e quais dados seriam visíveis para toda a organização (desempenho global das equipes e progresso global do sistema).

Como resultado, obteve-se uma organização parcialmente capacitada para utilização do Inspector. Na realidade, somente o membro envolvido em todos os passos foi completamente capacitado. Além disso, houve uma maior organização dos dados, de forma a garantir que a organização e seus membros tivessem visibilidade uniforme às informações de desempenho e progresso funcional, respeitando a privacidade de algumas informações.

6.3 Instanciando o Inspector

A atividade Instanciar o Inspector para o projeto Nota Fiscal Virtual não representou grande dificuldade já que, após a observação e o estudo do projeto,

verificou-se que ele possuía características que facilitavam a implantação do Inspector, pois os artefatos produzidos na organização são praticamente os mesmos daqueles definidos como padrão durante a definição de $\mu_{sistema}$, com pequenas variações já citadas anteriormente. Nesse momento, os conjuntos $ETAPAS$ e $A_j \mid j \in ETAPAS$, que indicam as etapas necessárias para o desenvolvimento de um caso de uso e os artefatos que devem ser produzidos durante a realização do projeto, respectivamente, foram configurados como segue:

- $ETAPAS$ = {especificação, análise e projeto, implementação, teste};
- $A_{especificação}$ = {documentação inicial, descrição funcional, fluxo de eventos};
- $A_{análise\ e\ projeto}$ = {diagrama de sequência, diagrama de classe, diagrama de estado, diagrama de atividade};
- $A_{implementação}$ = {diagrama de componente, código fonte, testes de unidade, versão executável do sistema}; e
- A_{teste} = {casos de teste, % de teste executados com sucesso}.

Após a definição dos conjuntos de artefatos que podem ser produzidos para desenvolvimento de um caso de uso no projeto, foi construído o artefato Critério de Avaliação dos Artefatos, onde foram documentados os critérios para inspeção destes conjuntos de artefatos. Os critérios utilizados para inspeção dos artefatos foram os mesmos critérios apresentados na Seção 4.3 como padrões.

Além de instanciar os conjuntos relevantes para o cálculo de $\mu_{sistema}$, foram definidas também as unidades de medida e regras de conversão usadas para padronizar e normatizar as métricas calculadas, os gráficos Gantt de atividades e os gráficos de linhas gerados. A unidade de tempo padrão utilizada foi *dias*, e as métricas calculadas tiveram precisão de duas casas decimais.

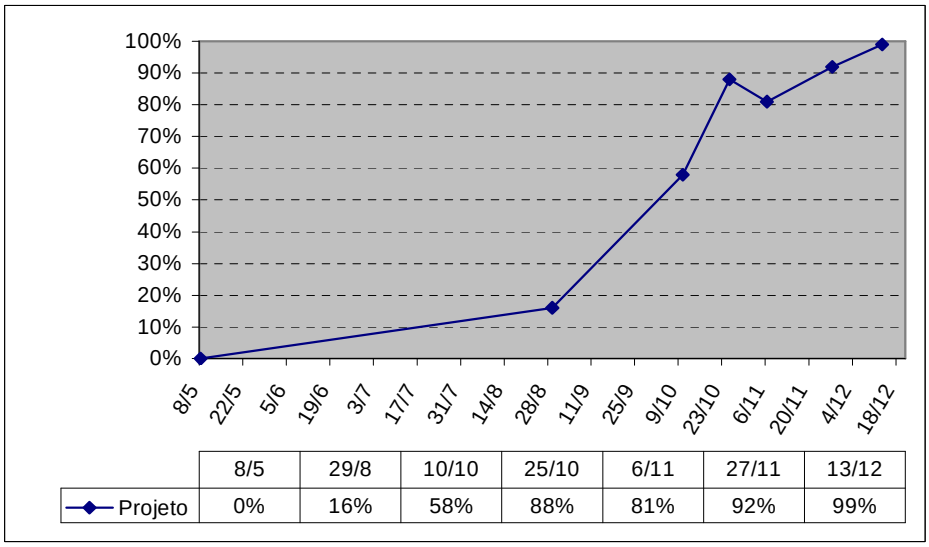
Outra preocupação que se teve durante a realização da atividade Instanciar o Inspector foi justamente definir os mecanismos de acesso às informações, ou seja, como os dados necessários para cálculo das métricas seriam acessados, e como os resultados

seriam mostrados. Basicamente, foi definido que os artefatos necessários para cálculo da métrica, exceto código fonte, teriam acesso compartilhado em uma homepage ou acesso via impressão das informações, e o código fonte seria acessado via rede interna da organização, diretamente na área de trabalho do desenvolvedor do caso de uso. Os resultados obtidos seriam informados via mensagens eletrônicas, papel impresso e|ou conversas informais.

6.4 Avaliando o Progresso

Foram realizadas cinco avaliações de progresso para o projeto Nota Fiscal Virtual, onde não foi possível recuperar a visão de desempenho, pois não existia nesse projeto a cultura de produção e atualização de cronogramas de atividades, sendo as datas definidas informalmente através de conversas entre os membros da equipe. Não existia de maneira bem definida também o papel do gerente de projeto, ficando essa responsabilidade compartilhada entre todos os membros da equipe de desenvolvimento. Desse modo, e devido ao fato de se ter iniciado o estudo de caso quando o projeto já tinha grande tempo de desenvolvimento, não houve um acompanhamento das métricas de desempenho, nem geração e atualização de gráficos PERT. A avaliação de progresso no projeto Nota Fiscal Virtual focalizou os aspectos funcionais incorporados ao sistema, ou seja, o acompanhamento de $\mu_{sistema}$ através da comparação dos valores alcançados com os esperados e da análise de tendências.

A Figura 6.2 representa a variação do progresso funcional do projeto Nota Fiscal Virtual durante o período em que seu desenvolvimento foi monitorado.

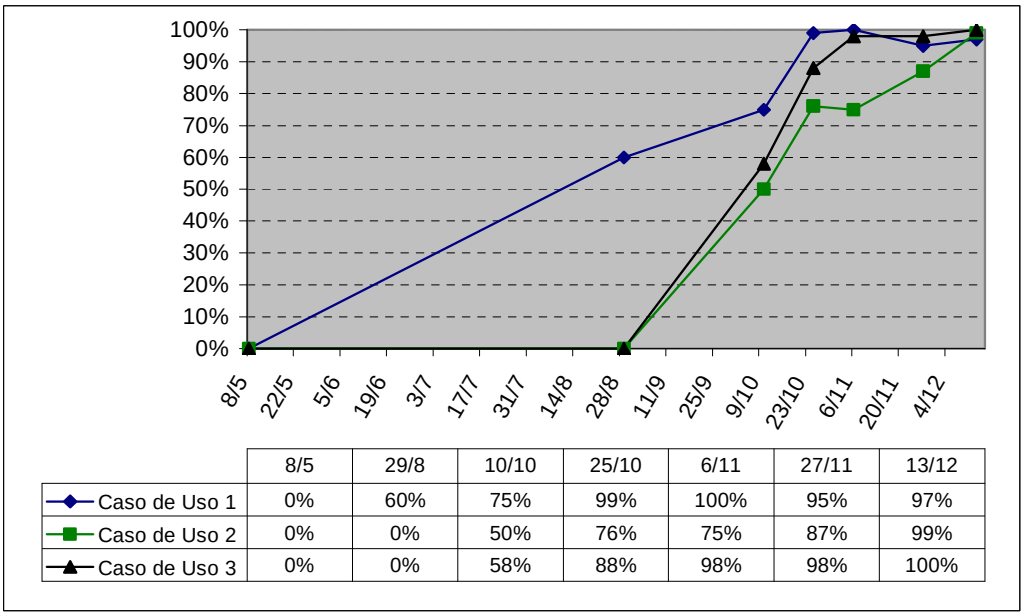


	8/5	29/8	10/10	25/10	6/11	27/11	13/12
Projeto	0%	16%	58%	88%	81%	92%	99%

Figura 6.2. Progresso do projeto durante as avaliacões realizadas.

Pode-se observar, ainda na Figura 6.2, que houve a tentativa de se recuperar o progresso do projeto no dia 29 de agosto, que corresponde a uma data anterior ao início do acompanhamento do projeto (iniciado no dia 28 de setembro). Os dados relativos ao progresso nessa data, foram obtidos através de conversas com os desenvolvedores e da recuperação de algumas informações registradas. Foi encontrado um valor próximo a 16%, resultado da realização de um único caso de uso, construído como exemplo durante o aprendizado do paradigma de desenvolvimento OO e da linguagem de programação Java.

Além do progresso do sistema como um todo, foi monitorado também o progresso de cada caso de uso relacionado ao projeto, onde procurou se observar problemas e dificuldades enfrentados no desenvolvimento de uma funcionalidade específica do sistema. Identificaram-se oito casos de uso que foram implementados para construção do sistema, sendo o *status* destes recuperados durante as cinco avaliações de progresso realizadas. A Figura 6.3 representa os resultados obtidos no acompanhamento de três destes casos de uso. Os resultados obtidos na avaliação dos demais casos de uso do projeto podem ser encontrados em [37].



	8/5	29/8	10/10	25/10	6/11	27/11	13/12
Caso de Uso 1	0%	60%	75%	99%	100%	95%	97%
Caso de Uso 2	0%	0%	50%	76%	75%	87%	99%
Caso de Uso 3	0%	0%	58%	88%	98%	98%	100%

Figura 6.3. Progresso dos casos de uso durante as avaliações realizadas

Antes de cada avaliação de progresso foi produzido o artefato denominado Plano de Avaliação do Progresso Técnico, onde foi documentado o planejamento da avaliação, indicando sua data, o escopo a ser considerado e o responsável pela coleta das informações. O membro da equipe alocado como Coletor de Informações foi sempre o mesmo em todas as avaliações e foi responsável por coletar informações sobre a equipe, que era única, e sobre o progresso funcional do projeto. As subseções seguintes apresentam, de forma sucinta, os resultados obtidos durante as cinco avaliações de progresso realizadas e o conteúdo do artefato Avaliação do Progresso Técnico produzido em cada uma destas avaliações.

6.4.1 Primeira Avaliação

A primeira avaliação de progresso realizada durante o estudo de caso ocorreu no dia 10 de outubro de 2000, onde o Coletor de Informações obteve como resultado $\mu_{sistema}(2) = 0{,}58$, ou seja, 58% das funcionalidades tinham sido incorporadas ao sistema. Observe que a primeira avaliação só se deu na segunda iteração $(i = 2)$, isso significa que o progresso já estava em andamento quando o estudo de caso se iniciou. Nesse momento, não havia informações suficientes que servissem como base para o planejamento dos valores esperados na avaliação, no entanto, a partir do tempo de

desenvolvimento já consumido (quase cinco meses) e de conversas informais com a equipe de desenvolvimento, estimou-se que o sistema deveria ter um progresso funcional acima de 60%. Desse modo, ficou constatado uma pequena defasagem em relação ao progresso esperado. Variações nesse pirmeiro resultado eram esperados, pois tratava-se da primeira aplicação do Inspector sobre o projeto, e não haviam dados históricos que permitissem uma estimativa precisa do progresso esperado.

Nessa avaliação observou-se a existência de apenas cinco casos de uso, podendo os resultados do progresso funcional encontrados para cada caso de uso e para o projeto como um todo serem observados na Tabela 6.1.

Caso de Uso	Especificação	Análise e Projeto	Implementação	Teste	Progresso Total
1	1	1	1	0	**0,75**
2	1	0,66	0,33	0	**0,50**
3	1	1	0,33	0	**0,58**
4	0,75	0,66	0,33	0,20	**0,49**
5	1	1	0,33	0	**0,58**
Projeto	**0,95**	**0,87**	**0,46**	**0,04**	**0,58**

Tabela 6.1. Progresso funcional obtido na primeira avaliação

Os casos de uso identificados pelos números 2 e 4 obtiveram progresso muito abaixo do esperado, indicando problemas na etapa de análise e projeto de ambos que, pelo tempo de desenvolvimento, já deveria estar finalizada. O caso de uso 4 apresentou problemas ainda maiores, pois nem sua especificação inicial tinha sido concluída. Foram encontrados, além do atraso nos casos de uso 2 e 4, problemas na utilização da metodologia que prega o desenvolvimento segundo um ciclo de vida iterativo, mas o que se observou foi um desenvolvimento sequencial, semelhante ao modelo de ciclo de vida cascata.

6.4.2 Segunda Avaliação

O atraso observado na primeira avaliação resultou em uma renegociação de datas, de forma que a nova data para finalização do projeto passou a ser o dia 6 de novembro. Desse modo, o planejamento da segunda avaliação baseou-se nessa nova data de finalização e nos resultados obtidos na primeira avaliação. Nesse momento, esperou-se um progresso significativo, cerca de 84% para o projeto como um todo, além de se

esperar incrementos significativos nos casos de uso que apresentaram problema na primeira avaliação.

O resultado obtido na segunda avaliação de progresso, realizada no dia 25 de outubro, foi de 88%, ou seja, $\mu_{sistema}(3) = 0{,}88$ como observado na Figura 6.2. Esse valor se enquadra dentro dos limites planejados para a avaliação, até mesmo superando o que estava previsto. Segundo a tendência observada, o projeto concluiria no tempo previsto, apesar de ser necessário um cuidado especial nos casos de uso 2 e 4, que ainda representavam grande risco. A Tabela 6.2 mostra os resultados obtidos nessa avaliação.

Caso de Uso	Especificação	Análise e Projeto	Implementação	Teste	Progresso Total
1	1	1	1	0,98	**0,99**
2	1	1	0,87	0,25	**0,78**
3	1	1	0,83	0,70	**0,88**
4	1	1	0,87	0,20	**0,77**
5	1	1	1	0,60	**0,90**
6	1	1	1	1	**1**
Projeto	**1**	**1**	**0,93**	**0,62**	**0,88**

Tabela 6.2. Progresso funcional obtido na segunda avaliação

O aparecimento de um sexto caso de uso nessa segunda avaliação de progresso revelou uma importante peculiaridade da métrica de progresso funcional. Tal peculiaridade relaciona-se com a sensibilidade da métrica, que considera o progresso funcional do projeto apenas em termos dos casos de uso atualmente levantados e definidos, não considerando o fato de que novos casos de uso poderão surgir e implicar no desenvolvimento de mais funcionalidades. Como consequência da observação desta peculiaridade de $\mu_{sistema}$, concluiu-se que sua utilização é mais apropriada quando os casos de uso do sistema a ser desenvolvido no projeto em questão já tiverem estabilizados, ou seja, quando quase não há mais o surgimento de novos casos de uso.

6.4.3 Terceira Avaliação

Na terceira avaliação, realizada no dia 06 de novembro, esperava-se obter valor próximo dos 100%, pois essa data coincidia com a data negociada para finalização do projeto. Mas ao contrário do que fora planejado, essa avaliação foi a que identificou os maiores problemas, apresentando um decréscimo nas funcionalidades do sistema que

caiu dos 88% obtidos na segunda avaliação para 81%, como observado na Figura 6.2. Esse decréscimo de 7% foi fruto do surgimento de novos requisitos em um estágio avançado do desenvolvimento e, além disso, os casos de uso que foram anteriormente avaliados como problemáticos (casos de uso 2 e 4) continuaram sem nenhum progresso, a espera de uma solução de desenvolvimento que precisou ser terceirizada. A Tabela 6.3 mostra o resultado obtido a partir da coleta e cálculo do progresso funcional nesta terceira avaliação.

Caso de Uso	Especificação	Análise e Projeto	Implementação	Teste	Progresso Total
1	1	1	1	1	**1**
2	1	1	0,50	0,50	**0,75**
3	1	1	1	0,90	**0,98**
4	1	1	0,50	0,50	**0,75**
5	1	1	1	1	**1**
6	1	1	1	1	**1**
7	0	0	0,62	0	**0,15**
8	1	0	0	0	**0,25**
Projeto	**0,90**	**0,86**	**0,77**	**0,70**	**0,81**

Tabela 6.3. Progresso funcional obtido na terceira avaliação

Os casos de uso 2 e 4 obtiveram decréscimo na etapa de implementação, pois foram encontrados erros que resultaram na necessidade de se incumbir a terceiros parte desse desenvolvimento. Além disso, os casos de uso 7 e 8 surgiram no escopo dessa avaliação, representando mais trabalho a ser realizado e, consequentemente, um decréscimo no progresso anterior. A partir dessa avaliação de progresso, observou a existência de problemas substanciais que ocasionaram um grande atraso no desenvolvimento. Além dos problemas enfrentados, estava prevista a realização de cursos de capacitação para os desenvolvedores que alocariam os mesmos, em tempo integral, durante 25 dias.

6.4.4 Quarta Avaliação

A quarta avaliação, realizada no dia 27 de novembro, apresentou um acréscimo significativo de funcionalidade que não era previsto devido à alocação do pessoal nos cursos de capacitação, obtendo $\mu_{sistema}(3) = 0{,}92$. Houve um incremento de 11% em relação à avaliação anterior, ou seja, o progresso funcional pulou de 81% para 92% nesta avaliação. Mesmo com esse incremento de funcionalidade, a tendência indicou

que o projeto finalizaria somente no fim do ano. Atrasos significativos já eram esperados pelo fato da organização estar em fase de transição, partindo para utilização de um novo paradigma e uma nova metodologia de desenvolvimento. A Tabela 6.4 mostra o resultado obtido na quarta avaliação.

Caso de Uso	Especificação	Análise e Projeto	Implementação	Teste	Progresso Total
1	1	1	1	0,80	**0,95**
2	1	1	0,80	0,70	**0,87**
3	1	1	1	0,90	**0,98**
4	1	1	0,85	0,80	**0,91**
5	1	1	1	1	**1**
6	1	1	1	1	**1**
7	1	1	1	1	**1**
8	1	0	0	0	**0,25**
Projeto	**1**	**0,95**	**0,90**	**0,84**	**0,92**

Tabela 6.4. Progresso funcional obtido na quarta avaliação

A Tabela 6.4 apresenta características interessantes como o decremento no progresso funcional do caso de uso 1, que se apresentava como resolvido na avaliação anterior e na avaliação atual mostra que ainda não está totalmente finalizado. Este decremento decorreu do surgimento de novas funcionalidades, requisitadas pelo usuário no momento do teste, que tiveram de ser incorporadas no caso de uso. Outra característica observada foi o completo desenvolvimento do caso de uso 7, que obteve um incremento de 85% em relação ao desenvolvimento anterior, chegando aos 100% de conclusão.

Ainda na quarta avaliação, observou-se também um incremento significativo nos casos de uso 2 e 4, que representavam maior risco, e verificou-se que tal incremento ocorreu devido a conclusão da parte dos mesmos, cujo desenvolvimento foi terceirizado, permitindo que o desenvolvimento destes, que estava paralisado, continuasse. Por fim, observou-se que o caso de uso que representava maior risco para o projeto naquele momento era o caso de uso 8, que ainda estava em estágio inicial de desenvolvimento, podendo apresentar alta complexidade e futuras complicações.

6.4.5 Quinta Avaliação

A quinta e última avaliação realizada para o projeto Nota Fiscal Virtual aconteceu no dia 13 de dezembro de 2000 e indicou o progresso esperado para finalização do projeto. O resultado encontrado foi $\mu_{sistema}(3) = 0{,}99$, ou seja, 99% das funcionalidades já tinham sido incorporadas ao sistema. Isso indica que o sistema se aproxima do final, estando em fase de implantação, onde testes de integração são realizados, alguns pequenos erros são corrigidos e os usuários validam a utilização do sistema. A Tabela 6.5 apresenta o resumo dos resultados obtidos na quinta avaliação.

Caso de Uso	Especificação	Análise e Projeto	Implementação	Teste	Progresso Total
1	1	1	0,95	0,95	**0,97**
2	1	1	0,98	0,98	**0,99**
3	1	1	1	1	**1**
4	1	1	1	1	**1**
5	1	1	1	1	**1**
6	1	1	1	1	**1**
7	1	1	1	1	**1**
Projeto	**1**	**1**	**0,98**	**0,98**	**0,99**

Tabela 6.5. Progresso funcional obtido na quinta avaliação

Como pode se observar, ainda resta alguma implementação e teste para serem realizados nos casos de uso 1 e 2, estando os demais finalizados. Outra característica importante que também pode ser observada na Tabela 6.5 é que o caso de uso 8 não foi considerado dentro dessa avaliação. O motivo de sua ausência foi o fato dele estar bastante atrasado e possuir complexidade relativamente alta, de modo que foi realizada uma reunião entre os membros da equipe e os usuários, onde ficou decidido que este caso de uso não seria mais desenvolvido nesta versão do sistema, sendo inserido somente em uma versão futura.

6.5 Conclusões do Estudo de Caso

Este estudo de caso serviu como fonte de auxílio para aplicação de parte do Inspector em projetos reais, sendo que a aplicação realizada sobre o projeto Nota Fiscal Virtual obteve resultados concretos e eficazes na detecção de problemas no desenvolvimento, através de atividades práticas que auxiliam o gerente de projeto na

tomada de decisões. Durante sua realização foram identificadas algumas falhas e limitações do Inspector, bem como observados o comportamento e a sensibilidade da métrica de progresso funcional.

A realização deste estudo de caso resultou também na modificação e adaptação de diversos conceitos e técnicas que estavam inicialmente inseridos no Inspector, mas que, quando aplicados na prática, mostraram algum tipo de deficiência. Por exemplo, no início da utilização do Inspector no projeto Nota Fiscal Virtual, a métrica $\mu_{sistema}$ era bastante rígida, sendo sempre os mesmos artefatos inspecionados para o seu cálculo, seja qual fosse o processo de desenvolvimento utilizado. Desse modo, durante a realização do estudo de caso, percebeu-se a necessidade de tornar a métrica mais adaptável para que ela pudesse ser utilizada sobre qualquer processo de desenvolvimento orientado a objetos que utilize conceitos de casos de uso, podendo ser configurada de acordo com os artefatos realmente produzidos para a realização de um caso de uso. Buscando superar essa limitação observada, foram realizadas modificações na métrica, onde foram inseridos os conjuntos $ETAPAS$ e A_{ij} que configuram a métrica para o processo de desenvolvimento utilizado na organização, indicando as etapas necessárias para o desenvolvimento de um caso de uso e o conjunto de artefatos que devem ser inspecionados em cada etapa para uma determinada iteração, respectivamente.

Esse estudo de caso permitiu também obter uma melhor visão da sensibilidade da métrica de progresso funcional, onde se percebeu que, muitas vezes, o progresso funcional de um determinado caso de uso e até mesmo do projeto diminuía, como pode ser observado através da comparação dos resultados obtidos na segunda (Tabela 6.2) e terceira (Tabela 6.3) avaliação. Isso, até certo ponto, representa um acontecimento estranho, pois como o progresso poderia retroceder? As pessoas imaginam que o projeto sempre tende a aumentar e que, consequentemente, isso indica progresso. O que se pode observar, no entanto, é que o tamanho do projeto não é diretamente proporcional ao progresso do mesmo. Por exemplo, um sistema pode estar maior (com mais linhas de código) do que quando realizada a avaliação anterior, mas devido ao surgimento de novos requisitos do usuário ou ao fato de existirem erros na produção de algum artefato

(que podem ser encontrados através da utilização de técnicas de inspeção que buscam observar a corretude das informações contidas nestes artefatos), o progresso do projeto como um todo poderá diminuir. Desse modo, ficou caracterizado durante a aplicação da métrica que ela considera o progresso em termos do esforço necessário para finalizar o projeto, observando o que foi feito e o que se falta fazer.

Outra conclusão tomada é que a utilização do Inspector depende, e muito, da participação e colaboração de todos os envolvidos no projeto, especialmente os analistas e programadores, que possuem conhecimento dos artefatos produzidos e das realizações dos casos de uso. Nesse estudo de caso, o Coletor de Informações era uma pessoa externa e não possuía conhecimento preciso do ambiente da organização. Nas duas primeiras avaliações o coletor recuperou as informações de maneira independente, sem envolver os membros da equipe. Já nas demais avaliações, a interação do coletor com os demais membros foi maior, havendo a participação dos mesmos no rastreamento e inspeção dos artefatos. Ficou claro que envolver a equipe de desenvolvimento na avaliação de progresso é essencial, pois o tempo gasto para realização das avaliações caiu pela metade quando houve a participação efetiva da equipe.

Por fim, percebeu-se que durante a aplicação do Inspector algumas limitações do processo de desenvolvimento podem atrapalhar, e até mesmo impedir, sua utilização eficaz. Por exemplo, a visão de desempenho não pôde ser recuperada no projeto Nota Fiscal Virtual, pois quando as avaliações de progresso começaram, o projeto já estava no meio (com quase cinco meses de desenvolvimento) e não estavam sendo produzidos cronogramas atualizados, planejando as atividades de desenvolvimento, que são a base para o monitoramento do desempenho.

6.6 Considerações Finais

A realização do estudo de caso que aplicou o Inspector a um projeto real foi muito importante para tornar clara as vantagens de se monitorar sistematicamente o desenvolvimento de um projeto. Foi possível identificar alguns problemas que, se não houvesse a utilização do Inspector, passariam desapercebidos e poderiam resultar no

fracasso do projeto. Alguns casos de uso foram identificados como problemáticos e puderam ser tratados e validados durante todo o desenvolvimento. Além disso, o Inspector permitiu também ao gerente de projeto verificar se os membros das equipes de desenvolvimento estavam realmente seguindo a metodologia empregada na organização, produzindo todos os artefatos previstos.

Foi possível perceber também que se torna bastante complexo identificar precisamente o progresso do projeto em organizações que não apresentam os pré-requisitos básicos para utilização do Inspector e, além disso, algumas limitações, tais como casos de uso não atualizados (que são escritos uma vez e nunca mais modificados) e projetos que se diferem completamente da descrição dos seus casos de uso, não permitem uma aplicação e identificação precisa do progresso funcional do sistema.

Durante o estágio de validação do Inspector, manteve-se contato com algumas organizações na tentativa de se aplicar o processo em mais de um projeto. Houve uma dessas organizações, além da Emprel, onde os contatos foram grandes, sendo realizadas várias reuniões e conversas com membros das equipes de desenvolvimento para introduzir o Inspector em um projeto em andamento. Assim, existiu uma aplicação inicial do Inspector sobre um subprojeto que estava sendo desenvolvido nesta organização, mas devido ao surgimento de algumas dificuldades internas na organização, o desenvolvimento deste subprojeto foi paralisado, não sendo possível completar a realização deste segundo estudo de caso. Apesar disso, foram realizadas as atividades iniciais previstas no Inspector, que correspondem às atividades Avaliar Status das Métricas na Organização e Instanciar o Inspector, onde foram produzidos os artefatos Visão Geral das Métricas na Organização e Critérios de Avaliação dos Artefatos. Houve também um único cálculo do desempenho de uma das equipes da organização, onde foram encontrados os valores de $\mu_{concluído}$, $\mu_{atraso\ médio}$ e $\mu_{novas\ atividades}$. Tais artefatos e resultados estão disponíveis em [37].

Capítulo 7

Conclusões

O trabalho realizado focalizou aspectos de gerenciamento que facilitam o acompanhamento de projetos de software, utilizando a manipulação de métricas como técnica para encontrar problemas no desenvolvimento e auxiliar o gerente na tomada de decisões.

Foi definido um conjunto de métricas de avaliação de progresso que, quando utilizadas com atenção, fornecem informações ao gerente sobre o progresso técnico do projeto e o desempenho das equipes envolvidas. Mas se observou que não basta um conjunto de métricas definidas e prontas para serem usadas, é necessário que a organização esteja disposta e preparada para coletar, calcular e avaliar tais métricas, identificando desvios nos valores que indiquem problemas no desenvolvimento e permitindo que o gerente identifique soluções para estes problemas [30].

Assim, foi criado um processo que visa fornecer o caminho que a organização deve seguir para utilizar, de maneira adequada, as métricas de progresso definidas. O Inspector representa, desse modo, uma tentativa de formalização das tarefas que devem ser realizadas para utilização sistemática de um conjunto de métricas de avaliação de progresso. Os responsáveis e os artefatos definidos são requisitos básicos para o gerenciamento de projetos dentro do processo. Eles foram agrupados e ordenados com o objetivo de melhorar a comunicação entre os membros da organização, garantindo que os resultados encontrados sejam condizentes com a realidade do projeto.

As seções seguintes apresentam algumas limitações observadas e perspectivas de trabalhos futuros que visam dar continuidade ao projeto realizado; trabalhos similares

realizados na área, servindo como fonte de consulta para pesquisa na área de processos para utilização de métricas de software; e por fim, apresenta algumas considerações finais sobre o trabalho realizado.

7.1 Trabalhos Futuros

Essa seção apresenta alguns trabalhos que podem ser realizados na área de pesquisa na qual esse trabalho foi centrado, indicando algumas possíveis melhorias no trabalho e complementações que podem torná-lo mais completo e adequado para o acompanhamento de projetos de software.

7.1.1 Superar Limitações da Métrica de Progresso Funcional

As limitações da métrica de progresso funcional, citadas na Seção 4.8, devem ser analisadas e superadas, ampliando o escopo da métrica e tratando algumas características não observadas como, por exemplo, a influência dos requisitos não funcionais [15] no desenvolvimento do sistema. Outra limitação que deve ser superada é tornar a utilização da métrica mais prática, indicando claramente como os pesos que fazem parte do cálculo da métrica podem ser derivados e definindo uma ferramenta que facilite seu cálculo. A Seção 7.1.4 descreve brevemente uma possível ferramenta que pode ser construída para semi-automatização do Inspector, incluindo o cálculo da métrica de progresso funcional.

7.1.2 Tratar os Pré-requisitos para Utilização do Inspector

Apesar de buscar ser facilmente aplicado, é notório que a inserção do Inspector implica em grandes modificações culturais dentro da organização e, consequentemente, sua utilização e adaptação não são tarefas triviais. Durante a definição do Inspector, alguns pré-requisitos foram considerados como premissas obrigatórias para utilização do mesmo, de forma a reduzir o escopo do trabalho realizado, e focalizando a recuperação do *status* de um determinado projeto.

Como já foi citado, para que visão de desempenho seja recuperada, foi assumido que a organização deve possuir uma certa preocupação com o planejamento, de forma que cronogramas sejam gerados indicando a duração esperada das atividades, e que sejam produzidos cronogramas realizados que representem a realidade atual, permitindo que o gerente compare o cronograma real com o estimado e, a partir disso, derive diretamente as métricas de desempenho definidas no processo.

Outro pré-requisito assumido para utilização do Inspector é que a organização apresente um certo controle dos artefatos produzidos durante o desenvolvimento, de forma que seja possível rastrear com facilidade o progresso dos casos de uso do sistema. É importante também que a organização tenha controle das versões dos artefatos produzidos, e que os membros das equipes obedeçam sistematicamente os padrões de documentação e armazenamento dos artefatos.

Assim, é interessante ampliar o Inspector de forma que ele indique um conjunto de atividades, passos e responsáveis, que buscam garantir que a organização obedeça a todos esses pré-requisitos, ou seja, é necessário definir tarefas sistemáticas que indiquem como tornar a organização capaz de manipular cronogramas, fazendo planejamento das atividades de desenvolvimento, alocando recursos e gerenciando se estas atividades estão sendo realizadas no tempo previsto. Além disso, devem ser incluídas no Inspector, tarefas que tornem o gerenciamento de versões e o rastreamento de artefatos um hábito dentro da organização, garantindo maior facilidade no rastreamento dos artefatos e cálculo das métricas.

7.1.3 Aumentar o Escopo do Inspector

O Inspector, como já dito anteriormente, representa um processo de auxílio ao gerenciamento de projetos que define um conjunto de atividades e métricas, que buscam medir o progresso do projeto, identificando atrasos e dificuldades técnicas no desenvolvimento de uma determinada funcionalidade. A partir dessa definição do Inspector, percebe-se que ele representa um processo com um objetivo bem específico: monitorar o desenvolvimento a partir da recuperação do progresso no desenvolvimento do projeto.

Organizações que utilizem o Inspector, apesar de adquirirem um avanço significativo em relação a organizações que não possuem monitoramento de progresso, ainda estão sujeitas a enfrentar problemas no desenvolvimento, bem como obter o insucesso do projeto devido à produção de um sistema que não captura as necessidades do cliente.

Desse modo, é interessante que o Inspector tenha seu escopo ampliado de forma a capturar todos os aspectos do gerenciamento de projetos, identificando não somente o andamento do projeto, mas observando também a qualidade do produto que está sendo desenvolvido, as relações entre os membros da equipe, as relações da organização com o cliente e preocupando-se em mapear riscos e características não técnicas que influenciam o progresso de um projeto, tais como conhecimento e experiência da equipe de desenvolvimento, complexidade do sistema, entre outros.

Outro trabalho interessante nesse aspecto seria adaptar o Inspector dentro de um processo de gerenciamento existente como, por exemplo, o fluxo de Planejamento e Gerenciamento do RUP [36], identificando onde as atividades e métricas do Inspector se encaixariam durante a realização das atividades de gerenciamento e planejamento do RUP. O resultado final seria um novo processo de gerenciamento, que contempla as características existentes no processo de gerenciamento original, inserindo nele a capacidade de acompanhamento sistemático do progresso, fornecida pelo Inspector.

7.1.4 Definição de uma Ferramenta de Apoio ao Inspector

Apesar de representar um importante auxílio para gerentes de projeto tornarem o processo de desenvolvimento mais controlável e mensurável, pode-se perceber que a aplicação do Inspector não é tarefa tão trivial. Ela exige grande dedicação, esforço e disciplina na coleta, cálculo e avaliação das métricas, tanto de desempenho quanto de progresso funcional. Muitas vezes, pode se tornar inviável para uma organização utilizar um processo de avaliação de progresso que exige um certo nível de maturidade da organização bem como tempo para aplicação do mesmo.

Desse modo, é interessante a construção de uma ferramenta que automatize ou, pelo menos, semi-automatize o Inspector, de forma que a coleta dos artefatos necessários para aquisição das métricas, os cálculos que devem ser realizados e a geração dos gráficos e tabelas resultantes da avaliação sejam realizados automaticamente, bastando ao usuário (no caso, o gerente de projeto) introduzir alguns parâmetros de configuração do processo, tais como o conjunto de artefatos e etapas necessários para desenvolvimento do caso de uso, os pesos relacionados a cada caso de uso, entre outros parâmetros, e, a partir disso, basta o gerente solicitar uma avaliação de progresso que a ferramenta produzirá os resultados que devem ser observados nessa avaliação. É importante salientar que, para o gerente de projeto configure corretamente o Inspector, é necessário que ele tenha grande conhecimento do processo de desenvolvimento utilizado pela organização.

A ferramenta deve ser configurável a um determinado ambiente de desenvolvimento, integrada a um conjunto de ferramentas pertencentes a esse ambiente, de forma a obter os dados necessários para cálculo e avaliação do progresso do projeto. Um exemplo de ambiente que a possível ferramenta poderia integrar seria um, no qual o projeto seria codificado em linguagem Java (através do uso do Visual Age [40]), utilizando o Rational Rose [46] como ferramenta para análise e projeto (modelagem UML), Microsoft Word [11] para documentação dos requisitos, Microsoft Project98 [45] para geração de cronogramas e planejamento de recursos e Sun JavaStar [53] para testes de rotinas a partir de uma interface gráfica (GUI). Assim, a ferramenta buscaria os artefatos de desenvolvimento produzidos em cada uma destas ferramentas do ambiente, inspecionando-os de modo a encontrar o desempenho das equipes e o progresso funcional do projeto.

Para recuperação do progresso funcional seriam identificados, a partir do arquivo do Rational Rose relativo ao projeto, os casos de uso que deveriam ser desenvolvidos no sistema. Em seguida, para cada caso de uso identificado, a ferramenta inspecionaria os artefatos relacionados com a realização deste caso de uso, verificando se os mesmos já teriam sido produzidos ou alterados de forma a conter as informações relativas ao desenvolvimento do caso de uso. A inspeção desses artefatos deve ser feita no ambiente

de desenvolvimento da organização, rastreando os mesmos a partir das ferramentas que integram o ambiente. Após a inspeção, o progresso do caso de uso é diretamente calculado. Feito isso para cada caso de uso, a ferramenta pode fazer o cálculo do progresso do sistema facilmente, através da Equação 4.3. De posse dos valores resultantes da avaliação do progresso das funcionalidades do sistema, a ferramenta produziria a Tabela Resumo do progresso funcional e os gráficos de progresso de cada caso de uso, e do sistema como um todo, fazendo uma projeção da evolução esperada para cada um destes.

A ferramenta buscaria o desempenho das equipes através da recuperação dos gráficos Gantt de atividades estimados e atuais, dos gráficos PERT de atividades atual e o obtido na avaliação anterior, produzidos no Microsoft Project98. Ela deve facilitar a análise dos dados por parte do gerente, fazendo a sobreposição automática do atual com o estimado, indicando, de alguma forma, as atividades em atraso, além de mostrar as diferenças entre o PERT atual e o anterior.

A partir do cronograma de atividades planejado e do atual (coletado pelo Coletor de Informações da equipe), temos que a ferramenta calcula automaticamente os valores das métricas de desempenho definidas, indicando se os valores obtidos são adequados ou se indicam problemas no desempenho das equipes. Para cada equipe, seria avaliado o quanto das atividades planejadas foram concluídas ($\mu_{concluído}$), quantas atividades não planejadas surgiram ($\mu_{novas\ atividades}$) e o atraso médio da equipe por atividade ($\mu_{atraso\ médio}$). De posse destes resultados, poderiam ser gerados os gráficos de desempenho de cada equipe, e gráficos do desempenho da organização como um todo, agrupando todos os projetos e equipes que a organização representa.

Apesar de ser possível automatizar grande parte do Inspector, existem partes do processo que podem enfrentar dificuldade para serem automatizadas, necessitando da presença e percepção humanas. Um exemplo de atividade que enfrentaria dificuldade para ser automatizada é Avaliar Status das Métricas na Organização, onde o Engenheiro de Processo deve estar em contato direto com os membros da organização, realizando conversas informais e reuniões, sendo essencial, nesse momento, a observação e o

feeeling do Engenheiro de Processo. Outro exemplo é a identificação dos problemas no desenvolvimento. A ferramenta pode auxiliar o gerente a identificar problemas através dos cálculos das métricas, mas a identificação precisa do motivo pelo qual o problema está acontecendo necessita de uma análise dos fatos e de conversas com a equipe onde o problema foi encontrado. Além disso, a solução destes problemas também envolve outros membros da equipe, depende da experiência do gerente de projeto e necessita de ponderações e observações que não podem ser recuperadas por um sistema de software. Uma tentativa de se amenizar estas dificuldades seria a definição de padrões para realização das atividades que envolvem fatores como experiência e comportamento humano, de modo que torne mais sistemática a aplicação do Inspector pelo gerente de projeto.

Apesar das limitações observadas, a construção de uma ferramenta que semi-automatize a realização das atividades e a produção dos artefatos definidos no Inspector, facilitaria, e muito, a aplicação do processo, tornando sua utilização uma tarefa mais simples e natural, facilitando a análise dos dados e a tomada de decisões.

7.2 Trabalhos Relacionados

Existem muitos trabalhos relacionados a essa área de pesquisa que, por ser uma necessidade real das grandes organizações no cenário atual, onde os sistemas estão cada vez maiores, mais complexos e envolvem uma quantidade maior de pessoas, representa uma área de grande interesse tanto científico quanto comercial. Essa seção apresenta alguns exemplos de trabalhos que estão sendo desenvolvidos como forma de auxílio ao gerenciamento de projetos.

Arnold e Pedross apresentam em [2], um método de medição implantado no *Swiss Banking Institute* que tem como objetivo avaliar a produtividade da organização, comparando seu desempenho com o de outras companhias, departamentos, e permitindo verificar a eficiência no uso de diferentes tecnologias. Este método, denominado *use case point* baseia-se na utilização de uma métrica que, assim como a métrica de progresso funcional definida no Inspector, é baseada em casos de uso. A idéia do *use*

case point é inspirada na análise por pontos de função [10], observando a funcionalidade especificada com casos de uso e cenários, e os fatores técnicos que são significativos para a funcionalidade que deve estar inserida no produto final. Apesar de se assemelhar em alguns pontos, existem grandes diferenças entre a métrica de progresso funcional definida no Inspector e o *use case point*, como por exemplo: o *use case point* não considera as etapas necessárias para o desenvolvimento de um caso de uso, nem pode ser configurado para se adequar a um determinado processo de desenvolvimento.

Jacquet e Abran [29] definem um modelo de processo para utilização de métodos de medição de software. O modelo proposto, de forma semelhante ao Inspector, detalha passos que devem ser seguidos para coletar e avaliar os resultados de métricas de software. A grande diferença é que este representa um modelo genérico de um processo de utilização de métricas, sendo, desse modo, independente da métrica que vai ser utilizada e do aspecto a ser focalizado (progresso, qualidade, etc.). Além disso, este modelo apresenta um interessante *framework* para analisar se uma métrica representa ou não os aspectos que ela se propõe a medir, ou seja, se a métrica é realmente válida.

Outro trabalho relacionado, também bastante interessante, é a proposta de um modelo de OOPM (*object oriented project management*) que emprega propriedades da orientação a objetos com a utilização de técnicas de redes de Petri [43], para fornecer uma especificação precisa das estruturas hierárquicas do processo de desenvolvimento de software [34]. Lin e Yeh definem, neste modelo, tipos de objetos que especificam as atividades do projeto e seus componentes através de representações textuais e gráficas. As propriedades OO de generalização-especialização e parte-todo são utilizadas de forma a representar as atividades em uma estrutura hierárquica. Como resultado, o modelo garante ao gerente um maior controle do processo de desenvolvimento, permitindo que ele recupere informações sobre o progresso do projeto em diferentes níveis na hierarquia de atividades.

Além dos trabalhos citados anteriormente, outros trabalhos relacionados podem ser encontrados em [14, 19, 30, 32, 41, 47, 51], buscando auxiliar o gerente de projeto no controle da qualidade do processo de desenvolvimento e do produto em construção.

7.3 Considerações Finais

Ainda que existam organizações preparadas, que apresentem um bom nível de maturidade e utilizem um processo de desenvolvimento de qualidade, a situação da maioria das empresas de desenvolvimento hoje não é das melhores. Tais organizações não mantém um padrão de qualidade na produção que controle seus produtos, satisfazendo os requisitos do cliente no tempo previsto e com um custo aceitável [33].

O sucesso de um projeto depende da eficiência da equipe de desenvolvimento e de um gerenciamento eficaz, que planeje as atividades, monitore os riscos e identifique dificuldades enfrentadas no desenvolvimento [44]. O trabalho realizado buscou o aperfeiçoamento do gerenciamento de projetos em organizações que realizam o desenvolvimento de software de médio e grande porte, introduzindo na organização uma cultura de utilização de métricas, manipulação e avaliação dos gráficos resultantes da recuperação das informações de progresso e produção sistemática de artefatos de gerenciamento, que visam melhorar a comunicação entre os membros da organização e servir como fonte de aprendizado, podendo ser reutilizados em projetos futuros.

Este trabalho abre espaço para realização de novos trabalhos em uma área ainda bastante nova e que necessita de pesquisa e incentivo, com o intuito de fornecer maior suporte para o processo de desenvolvimento de software. Um processo que formalize a utilização de um conjunto de métricas representa grande interesse para grandes empresas, que necessitam de um maior controle sobre seus projetos, e estão dispostas a investir para obter tais recursos.

O processo demonstrou ser aplicável no estudo de caso realizado (projeto Nota Fiscal Virtual), despertando o interesse de algumas empresas atuantes na área de tecnologia da informação em Pernambuco. Emprel, Procenge, Chesf e CSI acharam as contribuições do trabalho relevantes como forma de auxílio ao gerenciamento e demonstraram bastante interesse em aplicar o Inspector nos seus projetos. O estudo de caso realizado na Emprel (Projeto Nota Fiscal Virtual), serviu como um *feedback* para a organização avaliar a aplicação do Inspector nos projetos que ela realiza, a partir dos resultados positivos obtidos na sua aplicação, foi decidido que o processo será

implantado para todos os projetos desenvolvidos na mesma, inserindo o Inspector dentro da metodologia de desenvolvimento que está em fase de implantação na organização.

Uma versão prévia do trabalho realizado foi aceita e apresentada no Workshop de Qualidade de Software (WQS2000) realizado em outubro de 2000 [38]. Além disso, maiores detalhes do Inspector podem ser encontrados em sua versão estendida que está disponível em [37].

Referências

[1] Abreu F. B., Carapuça R. Candidate Metrics for Object-Oriented Software within a Taxonomy Framework, Journal of Systems Software, v. 26, n°1, p. 87-96, 1994.

[2] Arnold M., Pedross P., Software Size Measurement and Productivity Rating in a Large-Scale Software Development Department, IEEE, 0-8186-8368-6/98, 1998.

[3] Baker A. L., Bieman J. M., Fenton N., Gustafson D. A., Melton A., Whitty R., A Philosophy for Software Measurement, J. Systems and Software, 12, p. 277 - 281, 1990.

[4] Basili, V. R., Briand, L. C. and Melo, W. L. , IEEE Transactions on Software Engineering, A Validation of Object-Oriented Design Metrics as Quality Indicators. Volume 22, Number 10, pp 751-761, October, 1996.

[5] Baumert J. H., McWhinney M. S., Software Measures and the Capability Maturity Model, Documento No. CMU/SEI-TR-25, Carnegie Mellon University, Software Engineering Institute, 1992.

[6] Bohem B., Software Engineering Economics, 2.edition, Prentice-Hall, 1981.

[7] Booch G., Jacobson I., Rumbaugh J., The Unified Software Development Process, Addison-Wesley, 1999.

[8] Booch G., Jacobson I., Rumbaugh J., Unified Modeling Language – User's Guide, Addison-Wesley, 1999.

[9] Brito, Abreu F., MOOD – Metrics for Object-Oriented Design, OOPSLA'94 Workshop on Pragmatic and Theoretical Directions in Object-Oriented Software Metrics, 1994.

[10] Caldiera G., Antoniol G., Fiuten R., Lokan C., Definition and Experimental Evaluation of Function Points for Object-Oriented Systems, Proceedings of the 5th. International Symposium on Software Metrics, 1998.

[11] Camard B, Using Microsoft Word 2000, Special Edition, Que, maio, 1999.

[12] Card D. N., Glass R. L., Measuring Software Design Quality, Prentice Hall, 1990.

[13] Champeaux D., Object-Oriented Development Process and Metrics, Prentice Hall, 1997.

[14] Chidamber S.R and Kemerer C.F., A Metrics Suite for Object Oriented Design, IEEE Transactions on Software Engineering, vol.20, No. 6, Junho, 1994.

[15] Chung L., Representation and Utilization of Non-Functional Requirements for Information System Design, Department of Computer Science, University of Toronto, CAISE'91, 1991.

[16] Davies P., Brailsford T., New Frontiers of Learning – Guidelines for Multimedia Courseware Developers in Higher Education, disponível em UCoSDA (ISBN 1-85889-062-4), 1994.

[17] Empresa Municipal de Informática – Recife, Metodologia Emprel de Desenvolvimento de Software, Documento Interno, 2000.

[18] Fenton N. E., Pfleeger S. L.: Software Metrics – A Rigorous & Pratical Approach, Thomson Publications, 1997.

[19] Fowler P., Rifkin S., Software Engineering Process Group Guide, Software Engineering Institute, CMU/SEI-90-TR-24, ADA235784, 1990.

[20] Gilb T., Principles of Software Engineering Management, Addison-Wesley, 1990.

[21] Grady R. B., Pratical Software Metrics for Project Management and Process Improvement, Prentice-Hall, 1992.

[22] Harrison W., Magel K., Klunczny R., DeKock A., Applying Software Complexity Metrics to Program Maintenance, IEEE Computer, 15(9), p. 65-79.

[23] Henderson-Sellers B., Object Oriented Metrics: Measures of Complexity, Prentice Hall, 1996.

[24] Henderson-Sellers B., The Open Framework for Enhancing Productivity, IEEE Software, Vol. 17, No. 2, março/abril, 2000.

[25] Henderson-Sellers B., Younessi H., Graham I. S., The Open Process Specification, Addison Wesley, Open Series, 1997.

[26] Humphrey W. S., Introduction to the Personal software Process, SEI Series in Software Engineering, Addison-Wesley, 1997.

[27] Humphrey W. S., Managing the Software Process, Addison-Wesley, 1990.

[28] ISO/IEC 12207 Tecnologia da Informação – Processos de ciclo de vida de software.

[29] Jacquet J., Abran A., From Software Metrics to Software Measurement Methods: A Process Model, IEEE, 1082-3670/97, 1997.

[30] Joint Group on System Engineering, Pratical Software Measurement – A Foundation for Objective Project Management, Version 3.1a, disponível em http://www.psmsc.com, 1998. Último acesso em 02/2001.

[31] Jones C., Programming Languages Tables, Copyright by Software Productivity Research, disponível em http://www.spr.com/library/0langtbl.htm, 1997. Último acesso em 12/2000.

[32] Kirsopp C., Shepperd M., Webster S., An Empirical study into the use of measurement to support OO design evaluation, Empirical Software Engineering Research Group, ESERG TR99-01, 1999.

[33] Kotonia G., Sommerville I., Requirements Engineering Processes and Techniques, Horizon Pubs & Distributors Inc, 1998.

[34] Lin J., Yeh C., An Object-Oriented Formal Model for Software Project Management, Proceedings of the Asia Pacific Software Engineering Conference, 1998.

[35] Lorenz. M, Kidd J., Object-Oriented Software Metrics, Prentice Hall, 1994.

[36] Meneses J. B., Araújo A. C. M., Vasconcelos A. M. L., Moura H. P., ManageRUP - Uma Ferramenta Case para Gerência de Projetos de Software usando o Rational Unified Process, disponível em http://www.cin.ufpe.br/~jbm/RUP/CASE-RUP/ManagerUP.zip, 2000. Último acesso em 01/2001.

[37] Meneses J. B., Moura H. P., Inspector – Processo de Avaliação de Progresso de Projetos de Software Orientado a Objetos, disponível em http://www.cin.ufpe.br/~jbm/inspector, 2000. Último acesso em 01/2001.

[38] Meneses J. B., Moura H. P., Um Framework de Avaliação de Progresso de Projetos de Software Orientado a Objetos, Workshop de Qualidade de Software (WQS2000), João Pessoa-PB, outubro 2000.

[39] Minkiewiez A., Software Measurement: What's In It For Me?, Software Development Magazine, março, 2000.

[40] Nilsson D. R., Jakab P. M., Sarantakos B, Stinehour R. A., Enterprise Development with Visual Age for Java, Version 3, John Wiley & Sons, maio, 2000.

[41] Park E. R., Software Size Measurement: A Framework for Counting Source Statements, Documento No. CMU/SEI-TR-20, Carnegie Mellon University, Software Engineering Institute, 1992.

[42] Paulk M. C., Curtis B., Chrissis M. B., Weber C. V., Capability Maturity Model for Software (Version 1.1), Technical Report, Documento No. CMU/SEI-93-TR-024, Carnegie Mellon University, Software Engineering Institute, 1993.

[43] Peterson J., Petri Net Theory and The Modeling of Systems, Prentice-Hall, 1981.

[44] Pressman S. R., Software Engineering: A Practitioner's Approach, McGraw-Hill, 5ª ed., 1999.

[45] Pyron T., Using Microsoft Project 98, Special Edition, Que, outubro, 1997.

[46] Quatrani T., Visual Modeling with Rational Rose 2000 and UML, Addison-Wesley, 2ª ed., outubro, 1999.

[47] Queiroz E., Ambiente de Mensuração de Software Orientado a Objetos, Departamento de Ciências da Computação, Universidade de Brasília, Dissertação de Mestrado, 1999.

[48] Rational Software Corporation, Reaching CMM Levels 2 and 3 with the Rational Unified Process, disponível em http://www.rational.com/products/whitepapers/100416.jsp, setembro, 1998. Último acesso em 01/2001.

[49] Rothman J., Taking the Crunch Out of Crunch Time, Software Development Magazine, março, 2000.

[50] Rumbaugh J., Blaha M., Premerlani W., Eddy F., Lorensen W., Object Oriented Modeling and Design, Prentice Hall, 1991.

[51] Schroeder M., A Practical Guide to Object-Oriented Metrics, IT Professional - IEEE, Vol.1, No.6, novembro/dezembro, 1999.

[52] Solingen R., Berghout E., The Goal/Question/Metric Method: A Practical Guide for Quality Improvement of Software Development, McGrawn Hill, ISBN 0077095537, 1999.

[53] Sun Microsystems, JavaStar: the Java GUI Regression Testing Tool, disponível em http://www.sun.com/suntest/javastar, 2000. Último acesso em 01/2001.

[54] Travassos G. H., Shull F., Fredericks M., Basili V. R., Detecting Defects in Object Oriented Designs: Using Reading Techniques to increase Software Quality, Acm Sigplan Notices, USA, v.34, n.10, p.47-56, ISSN 0362-1340, 1999.

[55] Wiegers K., A Software Metrics Primer, Software Development Magazine, julho, 1999.

[56] Zuse H., Bollmann P., Software Metrics: Using Measurement Theory to Describe the Properties and Scales of Static Software Complexity Metrics, ACM SIGPLAN Notices, 24(8), p. 167-171, 1989.

Apêndice A

Modelos dos Artefatos do Inspector

Este apêndice apresenta o conjunto de artefatos produzidos do Inspector, cujos templates foram definidos durante a realização do trabalho, visando tornar mais prática e simples a aplicação e o preenchimento das informações necessárias durante a avaliação do progresso técnico. Os artefatos deverão ser preenchidos durante a aplicação do Inspector, de acordo com os passos definidos nas atividades. Estão disponíveis neste apêndice templates de todos os artefatos produzidos:

- Visão Geral das Métricas na Organização;
- Critérios de Avaliação dos Artefatos;
- Plano de Avaliação do Progresso Técnico;
- Modelo de Coleta de Informações sobre o Progresso do Projeto;
- Modelo de Coleta de Informações sobre o Progresso do Caso de Uso;
- Avaliação do Progresso Técnico; e
- Documento de Solução dos Problemas Identificados.

A notação utilizada para representar esses artefatos é bastante simples, sendo os artefatos subdivididos em seções, onde cada seção pode ser composta de subseções, e conter informações, representadas entre <>, que indicam como a seção deverá ser preenchida e fornecem dicas sobre como obter os dados necessários. As páginas seguintes apresentam os artefatos acima relacionados.

Visão Geral das Métricas na Organização

(Versão 1.0, 11 jan 2001)

1. Identificação

Nome da Organização: _____
Área de Atuação: _____
Localização: _____

_____

Responsável pelo Documento: _____
Produzido em: ___ / ___ / _____

2. Descrição Inicial da Organização

<Apresenta uma introdução ao documento, descrevendo rapidamente características gerais da organização, tais como: número de empregados, volume de negócios, tempo de existência, negócios realizados, etc.>

3. Objetivos

<Descreve os principais objetivos da avaliação do status da organização, indicando os fatores que foram observados durante o período de avaliação da organização e para que este artefato está sendo produzido.>

4. Situação Atual

<Documenta a situação do processo de desenvolvimento da organização e o resultado da avaliação da utilização de métricas na organização, focalizando aspectos de progresso.>

4.1. Processo de Desenvolvimento

4.1.1. Características do Processo

<Apresenta as características do processo de desenvolvimento atualmente utilizado na organização: paradigma de desenvolvimento, pessoas envolvidas, papéis definidos, atividades relacionadas à produção de software, ciclo de vida, etc. Aqui deve ficar clara a maturidade da organização e dos membros da equipe, bem como a tecnologia utilizada durante o desenvolvimento de projetos.>

4.1.2. Principais Problemas no Processo

<Descreve os principais problemas observados na organização durante o desenvolvimento de sistemas: problemas de comunicação entre membros da

equipe, falta de formalização do processo, deficiências técnicas, etc. A documentação destes problemas servirá como fonte de informação para implantação do Inspector, bem como observar a necessidade de se buscar novas soluções que melhorem o processo de desenvolvimento utilizado.>

<Indicar, quando possível, soluções que poderão ser aplicadas para adequar a organização a um processo de desenvolvimento mais completo e eficiente.>

4.2. Status da utilização de métricas

4.2.1. Métricas usadas na organização

<Indica a maturidade da organização quanto a utilização de métricas. Listar as métricas usadas, classificando-as, de acordo com o aspecto que ela captura (qualidade, progresso, tamanho, etc.)>

4.2.2. Problemas na utilização das métricas

<Lista de problemas relacionados à aplicação de métricas na organização: falta de estrutura para coleta de dados, métricas com eficiência não comprovada, mau uso dos resultados obtidos, não armazenamento dos resultados obtidos como forma de consulta a futuros projetos, etc.>

4.3. Fatores Internos

<Apresenta o ambiente de desenvolvimento em termos da comunicação do pessoal, local de trabalho, recursos existentes para suporte ao desenvolvimento, segurança e demais fatores organizacionais que podem influenciar no desenvolvimento de um sistema.>

4.4. Fatores Externos

<Identifica o perfil dos clientes da organização, os competidores, o que se pode com aprender deles, os outros stakeholders, fornecedores, parceiros, etc. Ou seja, apresenta todas as variáveis externas que influenciam no desenvolvimento dentro da organização.>

5. Artefatos produzidos atualmente

<Apresenta uma lista dos artefatos produzidos atualmente pela organização. A lista deve conter todos os artefatos que podem ser produzidos pela organização durante o desenvolvimento de um projeto. Para cada artefato da lista deve ser analisado quando o mesmo deve ser realmente produzido, se existem problemas na sua produção, e quais os motivos que impedem a sua produção.>

6. Conclusões

<Essa seção contém as conclusões sobre a Avaliação de Status da Organização, identificando as principais deficiências, classificando-as por ordem de prioridade, para que possam ser tratadas na atividade Adaptar a Organização.>

7. Referências

<Qualquer documento que foi utilizado como fonte de informação para produção do artefato Visão Geral das Métricas na Organização>

Critérios de Avaliação dos Artefatos
(Versão 1.0, 11 jan 2001)

1. Identificação

Nome do Projeto: _____
Breve Descrição do Projeto: _____
_____
_____
Responsável pelo Documento: _____
Produzido em: ___ / ___ / _____

2. Artefatos do Projeto

<Determina os artefatos que poderão ser produzidos durante o desenvolvimento do projeto e identifica quais os critérios para avaliar se os mesmos contêm as informações de um determinado caso de uso. Alguns artefatos padrões já estão inseridos no guia de Critérios para Inspeção dos Artefatos. Se o artefato inserido no guia for produzido nesse projeto, basta copiar o item relacionado para esse documento (ou fazer referência para o guia), caso contrário, os critérios deverão ser definidos e documentados, além de armazenados e incluídos na guia como forma de futura referência.>

2.1. <Nome do Artefato>

<Lista os critérios para avaliar se o artefato foi alterado ou criado para conter as informações relativas à realização das etapas de desenvolvimento de um determinado caso de uso.>

<Deve ser criado um item deste para cada artefato que poderá ser desenvolvido durante a realização de um caso de uso no projeto.>

3. Tabela de Artefatos por Etapa

<Tabela contendo as etapas para realização do caso de uso e seus respectivos artefatos. Identifica o conjunto de aretefatos A_{ij} para cada etapa definida.>

Etapa	Identificação do Conjunto	Artefatos que devem ser inspecionados
Especificação	A_1	Diagrama de Caso de Uso Documento de Requisitos ...
...	A_2	...
...	A_3	...

4. Referências

<Qualquer documento que foi utilizado como fonte de informação para produção do artefato Critério de Inspeção dos Artefatos.>

Plano de Avaliação do Progresso Técnico

(Versão 1.0, 11 jan 2001)

1. Identificação

Nome do Projeto: _____

Breve Descrição do Projeto: _____

_____

_____

Responsável pelo Plano: _____

Produzido em: ___ / ___ / _____

2. Escopo

<Determina o escopo da avaliação de progresso a ser realizada. Identifica os casos de uso e atividades focalizados nessa avaliação, bem como o intervalo de tempo de realização do projeto que será considerado.>

3. Metas da Avaliação

<Essa seção indica quais são os resultados esperados durante a avaliação de progresso. Tais metas são definidas a partir das informações contidas em artefatos de apoio (Plano de Projeto, Plano de Iteração), da análise de tendências da avaliação anterior, além de experiências anteriores com projetos semelhantes.>

4. Datas e Responsabilidades

<Essa seção consiste em uma tabela, onde cada linha define a equipe que será avaliada, o coletor de informações (responsável pela coleta dos dados relativos a uma equipe), os casos de uso que ele deverá coletar as informações de progresso funcional e a data que a coleta deverá ser realizada. Tais informações serão usadas no preenchimento do artefato Modelo de Coleta de Informações sobre o Progresso do Caso de Uso.>

Coletor	Equipe a ser avaliada	Casos de Uso Focalizados	Data da Coleta
Fulano	Equipe X	Solicitar Pedido Efetuar Compra	15/10/00
...	...	...	...
...	...	...	...

5. Referências

<Qualquer documento que foi utilizado como fonte de informação para produção do Plano de Avaliação do Progresso Técnico.>

Modelo de Coleta de Informação de Progresso sobre o Projeto

(Versão 1.0, 11 jan 2001)

Identificação

Nome da Organização: _____
Tipo de Aplicação: _____
Nome do Projeto: _____
Breve Descrição do Projeto: _____
_____
_____
Gerente do Projeto: _____
Número de Equipes do Projeto: _____
Data de Avaliação: ___ / ___ / _____
Data Avaliação Anterior: ___ / ___ / _____ Estado Anterior: _____

Estado dos Casos de Uso do Projeto

Número Total de Casos de Uso: _____

Identificação do Caso de Uso	Prioridade do Caso de Uso	Equipe Responsável	Estado ($\mu_{caso\ de\ uso}$)

Estado do Projeto

Estado do Projeto ($\mu_{projeto}$) : _____
Incremento Percentual: _____

	SIM	NÃO
Resultado Esperado?	☐	☐

Comentários:

Modelo de Coleta de Informação de Progresso sobre o Caso de Uso

(Versão 1.0, 11 jan 2001)

Identificação

Nome do Projeto: _____
Número de Identificação do Caso de Uso: _____
Nome do Caso de Uso: _____
Breve Descrição do Caso de Uso: _____
_____
_____
Responsável pelo Caso de Uso: _____
Data de Avaliação: ___ / ___ / _____ Responsável pela Coleta: _____
Data Avaliação Anterior: ___ / ___ / _____ Estado Anterior: _____

Estado da Especificação

- **Documentação Inicial**

	CONCLUÍDO	NÃO CONCLUÍDO
Documentação Inicial:	☐	☐

$\mu_{documentacaoinical}$ = _____

- **Descrição Funcional + Fluxo de Eventos**

Descrição Funcional

	CONCLUÍDO	NÃO CONCLUÍDO	NÃO NECESSÁRIO
Descrição do Caso de Uso	☐	☐	☐
Entradas Definidas	☐	☐	☐
Saídas Definidas	☐	☐	☐
Pré-Requisitos	☐	☐	☐
Pós-Requisitos	☐	☐	☐

$\mu_{descricaofuncional}$ = _____

Fluxo de Eventos

	CONCLUÍDO	NÃO CONCLUÍDO	NÃO NECESSÁRIO
Fluxo Principal	☐	☐	☐
Fluxo Alternativo	☐	☐	☐
Subfluxos	☐	☐	☐

$\mu_{fluxodeeventos}$ = _____

- **Interface Prototipada**

	CONCLUÍDO	NÃO CONCLUÍDO	NÃO NECESSÁRIO
Especificação da Interface	☐	☐	☐
Projeto de Interface	☐	☐	☐
Implementação do Protótipo	☐	☐	☐

$\mu_{\text{int}erface}$ = _____

	TOTAL	PARCIAL	NENHUMA
Realização da Especificação:	☐	☐	☐

$\mu_{especificacao}$ = _____

Estado da Análise e Projeto

Diagramas UML:	REFLETE O CASO DE USO	NÃO REFLETE O CASO DE USO	NÃO NECESSÁRIO
Sequência/Colaboração	☐	☐	☐
Classe	☐	☐	☐
Estado	☐	☐	☐
Atividade	☐	☐	☐

	TOTAL	PARCIAL	NENHUMA
Realização da Análise + Projeto:	☐	☐	☐

$\mu_{ana+prj}$ = _____

Estado da Implementação

	CONCLUÍDO	NÃO CONCLUÍDO	PARCIALMENTE CONCLUÍDO	NÃO NECESSÁRIO
Diagrama de Componentes	☐	☐	☐ _____ %	☐
Implementação das Classes Relacionadas	☐	☐	☐ _____ %	☐
Interface com Funcionalidades Disponíveis	☐	☐	☐ _____ %	☐
Testes de Unidade das Classes	☐	☐	☐ _____ %	☐

Realização da Implementação:	TOTAL	PARCIAL	NENHUMA
	☐	☐	☐

$\mu_{implementacao}$ = _____

Estado do Teste

	CONCLUÍDO	NÃO CONCLUÍDO	NÃO NECESSÁRIO
Plano de Teste do Caso de Uso	☐	☐	☐
Projeto de Teste do Caso de Uso	☐	☐	☐
Implementação do Teste do Caso de Uso	☐	☐	☐

% dos testes do caso de uso realizados com sucesso = _____

Realização dos Testes:	TOTAL	PARCIAL	NENHUMA
	☐	☐	☐

μ_{teste} = _____

	TOTAL	PARCIAL	NENHUMA
Realização do Caso de Uso:	☐	☐	☐

Etapas Realizadas: ☐

- Especificação ☐
- Análise + Projeto ☐
- Implementação ☐
- Teste ☐

Estado do Caso de uso ($\mu_{casodeuso}$) : _____

Incremento Percentual: _____

Comentários:

Avaliação do Progresso Técnico
(Versão 1.0, 11 jan 2001)

1. Identificação

Nome do Projeto: _____
Breve Descrição do Projeto: _____
_____
_____
Responsável pela Avaliação: _____
Data de Avaliação: ___ / ___ / _____

2. Introdução

<Faz uma introdução ao documento de avaliação, identificando porque ele é produzido e qual sua importância. Além disso, informa o momento da avaliação de progresso, fazendo referência ao escopo e às metas definidas no Plano de Avaliação do Progresso Técnico.>

3. Desempenho das Equipes

<A subseção abaixo deverá ser criada para cada equipe de desenvolvimento pertencente ao projeto a ser avaliado, mostrando o desempenho da mesma através da comparação dos gráficos de atividades estimados e atuais, e do cálculo das métricas de desempenho definidas.>

3.1. <Nome da Equipe>

<Deverá conter os gráficos de atividades (Gantt) atuais e planejados de cada equipe, para que se possa observar as atividades com atraso relativamente grande.>

<Deverá conter o gráfico PERT atualizado, com dados relativos às atividades da equipe em consideração, contendo as dependências entre as atividades e o caminho crítico de atividades.>

<Atrasos relevantes deverão ser identificados em uma tabela como a mostrada abaixo.>

Atividade	Atraso Identificado	Motivo
Atividade X	10 dias	Realocação de pessoal para resolução de problemas de emergência
...	...	...

<Os valores resultantes do cálculo do desempenho das equipes, ou seja, os valores de $\mu_{concluído}$, $\mu_{atraso\ médio}$, $\mu_{novas\ atividades}$. A avaliação dos resultados obtidos destas métricas também dever colocado aqui, identificando problemas de atraso, dificuldades de planejamento e equipe com um grande atraso relativo por atividade.>

4. Progresso Funcional

<Essa seção preocupa-se em documentar o progresso funcional dos casos de uso pertencente ao sistema, devem ser salientadas variações inesperadas no progresso das funcionalidades, tanto positivamente quanto negativamente.>

4.1. Tabela de Progresso Funcional

<Resume em uma tabela as informações relativas ao progresso nas funcionalidades de cada caso de uso, bem como as informações do progresso funcional do projeto. Considera também o progresso de cada etapa para o desenvolvimento do caso de uso e do projeto. A tabela abaixo consiste em um exemplo de Tabela Resumo.>

Caso de Uso	Especificação Inicial	Análise e Projeto	Implementação	Teste	Progresso Total
1	0,66	0,66	0	0	**0,33**
2	1	1	0,66	0,50	**0,79**
3	0,75	0,66	0	0	**0,35**
4	0,75	0,66	0	0	**0,35**
5	1	1	1	1	**1**
Projeto	**0,94**	**0,92**	**0,64**	**0,58**	**0,77**

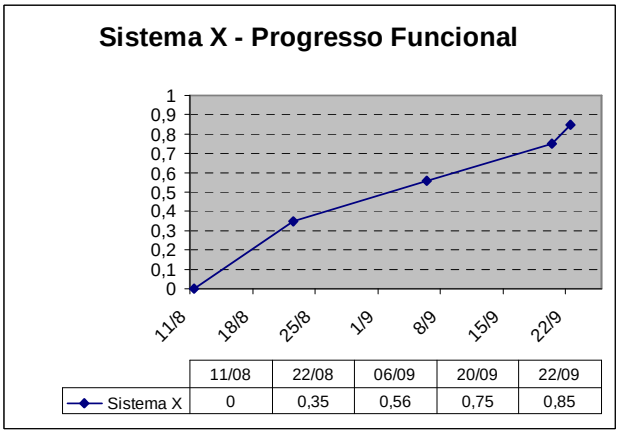
<Deverá conter a documentação e discussão dos dados mostrados na Tabela Resumo, identificando as principais variações de progresso observadas em relação à tabela gerada na avaliação anterior (se houver). Aqui, poderá ser criada uma tabela auxiliar, contendo informações sobre o incremento de funcionalidade obtido por cada caso de uso, e pelo projeto, desde a última avaliação.>

4.2. Gráficos de Progresso Funcional

<Deverá conter uma subseção para o projeto e uma para cada caso de uso pertencente ao projeto.>

4.2.1. Progresso Funcional do <Nome do Projeto>

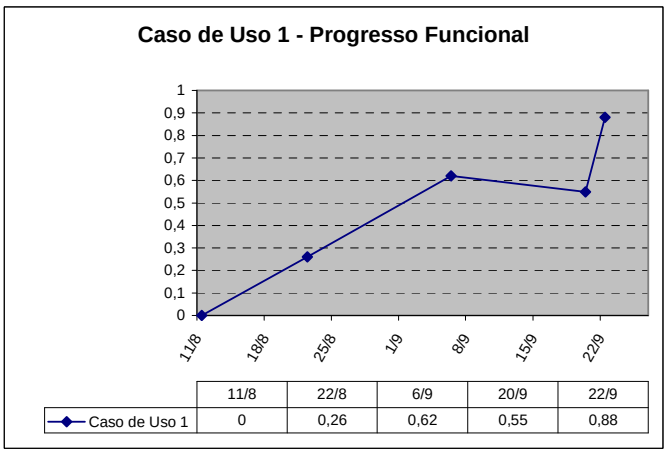
<Deverá ser inserido o gráfico de linha descrevendo o progresso funcional do projeto em relação ao tempo, poderão ser realizadas análise de tendências sobre o mesmo. Abaixo segue um exemplo de gráfico de linha com informações sobre o progresso do projeto.>



<Documentação da análise do gráfico representado. Deverão ser descritos possíveis problemas identificados através da observação de progresso e/ou tendência não esperado.>

4.2.2. Progresso Funcional do <Nome do Caso de Uso>

<Gráfico de linha descrevendo o progresso funcional do caso de uso em relação ao tempo, realizando análise de tendências sobre o mesmo. Abaixo segue um exmplo de gráfico de linha com informações sobre o progresso de um caso de uso.>



<Documentação da análise do gráfico representado. Deverão ser descritas as grandes variações de valores de progresso funcional identificados no desenvolvimento do caso de uso, pois esse tipo de variação indica que algum problema possivelmente está acontecendo na realização do caso de uso.>

5. Problemas Identificados

<Essa seção contém um resumo de todos os problemas e riscos problemáticos identificados durante a avaliação do progresso técnico do projeto. Estes problemas serão colocados de forma breve em uma tabela, de modo a facilitar a leitura e entendimento dos memos.>

Problema	Descrição do Problema
Problema X	Atraso exagerado no caso de uso 1 devido a sua complexidade e falta de conhecimento da equipe com a tecnologia necessária.
...	...
...	...
...	...

6. Conclusões

<Documentação das conclusões retiradas a partir dessa avaliação, listando as dificuldades enfrentadas durante a recuperação das informações necessárias para avaliação de progresso. Visão geral dos principais problemas encontrados, e avaliação inicial do impacto que esses problemas podem causar se não forem tratados.>

7. Referências

<Qualquer documento que foi utilizado como fonte de informação, para produção do artefato Avaliação do Progresso Técnico.>

Documento de Solução dos Problemas Identificados

(Versão 1.0, 11 jan 2001)

1. Identificação

Nome do Projeto: _____

Breve Descrição do Projeto: _____

_____

_____

Responsável pela análise dos problemas: _____

Data de Avaliação: ___ / ___ / _____

2. Introdução

<Identifica os objetivos deste artefato e o relaciona aos principais documentos relativos à avaliação realizada (Plano de Avaliação do Progresso Técnico, Modelo de Coleta de Informação sobre o Progresso do Projeto, Modelo de Coleta de Informação sobre o Progresso do Caso de Uso e Avaliação do Progresso Técnico). Faz referência ao escopo da avaliação realizada e aos principais problemas identificados.>

3. Problemas Identificados

<Para cada problema deverá ser criada uma subseção descrevendo o problema, identificando possíveis soluções e escolhendo a solução mais adequada>

3.1. <Problema Identificado>

<Deverá conter uma descrição completa do problema identificado após a análise da avaliação de progresso realizada e de conversas com pessoas chave envolvidas com o problema em questão.>

<Deverá conter uma subseção para cada possível solução do problema descrito. Além disso, deverá conter uma subseção que deverá documentar a escolha da solução mais adequada, mostrando os motivos pelo qual essa solução foi escolhida.>

3.1.1. Solução Alternativa <Número de Identificação da Solução>

<Uma solução alternativa para o problema deverá ser apresentada nesse momento, contendo uma descrição da solução, das pessoas envolvidas com a solução, do esforço necessário, custos envolvidos e o impacto que a aplicação dessa solução representará no desenvolvimento.>

3.1.2. Escolha da Solução

<A solução mais adequada será escolhida através da criação de uma tabela simples, contendo o mapeamento das soluções aos seus respectivos parâmetros de avaliação, ou seja, aos fatores que são relevantes para escolha da melhor solução para a organização. A tabela abaixo representa um exemplo de tabela de auxílio para tomada de decisões.>

Alternativa de Solução	Facilidade de Implantação	Confiabilidade	Efetividade	Cobertura	Riscos não problemáticos
1		✓		✓	
2	✓	✓	✓		✓
3	✓		✓		✓
4		✓	✓		

<Após análise das possíveis soluções, a alternativa que representar o menor risco para o desenvolvimento deverá ser escolhida e implantada. A solução escolhida deverá ser indicada nesse momento, indicando também quais os aspectos mais importantes para sua escolha.>

4. Resultados Obtidos

<Documenta o *status* dos problemas identificados após a implantação da solução escolhida para cada problema. Deverá conter uma subseção para cada problema identificado.>

4.1. Status do Problema: < Problema Identificado>

<Apresenta os resultados da aplicação da solução escolhida para solução do problema. Quais os problemas enfrentados durante esse processo de solução, o impacto para o desenvolvimento (se representou atraso relevante para o projeto) e se o problema foi completamente sanado.>

<Além disso, temos que os artefatos de apoio ao desenvolvimento como: plano de projeto, plano de iteração, etc., devem ser atualizados para refletir a atual situação do projeto após a solução do problema.>

5. Referências

<Qualquer documento que foi utilizado como fonte de informação para produção do artefato Documento de Solução dos Problemas Identificados.>