Capítulo
9

Gerenciamento de Projeto de Software

Nos capítulos anteriores foi visto como planejar um projeto, e atenção especial foi dada aos processos de estimação de esforço e controle de riscos. Este capítulo trata do momento em que o projeto efetivamente se inicia. Alguns aspectos da gerência de um projeto já foram discutidos nos capítulos anteriores, mas aqui é aprofundado o processo de gerenciamento como um todo. Inicialmente, é discutido o papel do *gerente de projeto* (Seção 9.1) e suas atribuições, além de recomendações práticas sobre como ele deve atuar. Em seguida, são apresentados rapidamente os conceitos de gerenciamento de projetos de uma fonte internacionalmente reconhecida, o *Project Management Book of Knowledge*, também conhecido como PMBOK (Seção 9.2). Depois, são apresentados conceitos de outro método de gerenciamento também bastante usado, o PRINCE2, ou *Projects in a Controlled Environment* 2 (Seção 9.3). Além disso, são apresentadas recomendações gerais sobre como *conduzir* um projeto de software (Seção 9.4), *métricas* a serem usadas (Seção 9.5), como *revisar e avaliar* (Seção 9.6) o andamento de um projeto e como realizar o *fechamento* ou conclusão de um projeto (Seção 9.7).

A gerência de projeto pode ser entendida como uma disciplina dentro de um processo de engenharia de software, em geral exercida por um único indivíduo (o gerente de projeto) cuja função é levar o projeto a alcançar os objetivos planejados dentro dos prazos, com o custo e a qualidade previstos. Para ter sucesso, o gerente deve manter os riscos do projeto nos níveis de probabilidade e impacto mais baixos possível, avaliando continuamente o progresso e adotando medidas proativas para a redução desses riscos ou medidas de correção se, apesar de tudo, houver problemas.

Uma fonte de referência importante sobre a gerência de projetos é o PMBOK (PMI, 2017). Porém, como ele é um referencial genérico para a condução de projetos, é necessário ter em mente que projetos na área de software têm suas particularidades. Assim, um gerente de projeto de software precisa ter conhecimentos específicos sobre gerência de projetos *de software*.

O SWEBOK (IEEE Computer Society, 2014) indica, entre outras coisas, as seguintes particularidades para os projetos de software:

- Os clientes dificilmente percebem as reais complexidades envolvidas no processo de desenvolvimento de software, em especial as relacionadas com as mudanças de requisitos.
- É praticamente inevitável que os próprios processos de engenharia de software acabem gerando a necessidade de introdução de novos requisitos ou de modificação dos requisitos existentes.
- Como resultado disso, o software é frequentemente construído por um processo de refinamento interativo em vez de uma sequência de atividades previamente bem definidas e programadas.
- A engenharia de software necessariamente incorpora aspectos de criatividade e disciplina. Manter um balanceamento apropriado entre esses dois aspectos costuma ser uma tarefa difícil.
- Em geral, o grau de novidade e de complexidade do software é extremamente alto.
- A tecnologia subjacente ao software muda com muita frequência.

O SWEBOK divide a área de gerenciamento de projeto em engenharia de software nas seguintes subáreas:

- Inicialização e definição de escopo.

- Planejamento de projeto de software.
- Condução de projeto de software.
- Revisão e avaliação.
- Fechamento.
- Medição em engenharia de software.

As primeiras duas subáreas são tratadas no Capítulo 6. Este capítulo vai abordar as outras subáreas: condução de um projeto de software, sua revisão, medição, avaliação e fechamento.

9.1 Gerente de Projeto

Um projeto bem-sucedido depende muito de um bom gerente. Isso vale tanto em situações nas quais existe a figura do gerente como indivíduo quanto nas situações em que a equipe se auto gerencia. De uma forma ou de outra, o processo de planejamento, condução, avaliação e fechamento do projeto estará presente. Várias habilidades podem ser observadas em bons gerentes de projeto de software, entre elas:

- *Têm o dom de prever*: bons gerentes têm a capacidade de visualizar e antecipar problemas antes que eles ocorram, e tomar ações preventivas.
- *São organizados*: a organização tem vários aspectos, mas um que parece ser especialmente importante para o gerente de projeto é a capacidade de visualizar prioridades e passá-las para sua equipe. Assim, em meio à complexidade de um projeto, o gerente será capaz de enxergar o que é realmente importante e concentrar os esforços da equipe nisso.
- *Sabem liderar*: o gerente de projeto não é apenas um chefe. Além da equipe, ele precisa interagir com pessoas que não estão sob seu comando (clientes, usuários e especialistas de domínio, por exemplo). Assim, ele precisa ter o carisma de líder e ser capaz de motivar as pessoas a gastarem seu tempo nas atividades que são necessárias para o projeto.
- *São bons comunicadores*: eles são capazes de utilizar múltiplos meios de comunicação, como *e-mail*, telefone, reuniões, apresentações etc., para obter e transmitir as informações necessárias. Eles são efetivos, objetivos e pragmáticos quando se comunicam (não ficam fazendo rodeios). Além disso, são capazes de ouvir. Um gerente que não ouve sua equipe ou outros interessados poderá levar um projeto a fracassar, pois pequenos problemas não resolvidos de início muitas vezes acabam virando grandes problemas no final de um projeto de desenvolvimento de software.
- *São pragmáticos*: existem dois extremos quando se pensa na forma de tomada de decisão de um gerente: os que adiam a decisão até conhecer todas as implicações das opções e os que tomam decisões por impulso, sem pensar. No equilíbrio entre esses dois extremos está o gerente pragmático: ele analisa os prós e contras da forma mais eficiente possível, e é capaz de avaliar rapidamente se está em condições de tomar uma decisão fundamentada ou não. Contudo, caso a análise dos prós e contras tome mais tempo do que tentar uma das opções, ele vai perceber isso e decidir se tenta um caminho ou o outro.
- *São empáticos*: um gerente não faz o trabalho da equipe nem faz seu trabalho sozinho. Ele precisa se apoiar em outras pessoas. Para obter essa colaboração, ele precisa entender o que motiva as pessoas, ter empatia, que é a capacidade de colocar-se no lugar do outro e entender suas necessidades. A partir disso, o gerente balizará suas ações de motivação.

Classicamente, assumia-se que o gerente de projeto precisava equilibrar os três vértices do *triângulo de restrições*, que envolvia *tempo*, *custo* e *escopo*. Diminuir qualquer um desses ângulos provocaria aumento em pelo menos um dos outros dois. Porém, hoje esse triângulo é substituído

por um conjunto de seis variáveis: *escopo*, *qualidade*, *cronograma*, *orçamento*, *recursos* e *riscos*. Assume-se ainda que qualquer mudança em um desses elementos afetará pelo menos um dos outros.

9.2 Gerenciamento de Projetos segundo o PMBOK

Apesar de não ser específico para projetos de software, o PMBOK (PMI, 2017) é uma excelente referência em termos de gerenciamento de projetos. Ele estrutura o corpo de conhecimentos em duas dimensões: *grupos de processo* e *áreas de conhecimento*.

Os grupos de processo PMBOK são iniciação, planejamento, execução, monitoramento e controle e encerramento. Estes grupos equivalem mais ou menos às fases de gerenciamento identificadas no SWEBOK.

As áreas de conhecimento do PMBOK são descritas na Tabela 9.1. As áreas de conhecimento não abrangem necessariamente todos os grupos de processo; apenas o gerenciamento de integração faz isso.

Tabela 9.1 Áreas de conhecimento de gerência de projetos segundo o PMBOK

Área de gerenciamento	**Descrição**
Integração	Atividades que o gerente de projetos executa de forma a garantir que todas as partes do projeto funcionem juntas.
Escopo	Atividades necessárias para que o projeto execute de fato o que for preciso para gerar o produto e *somente* isso.
Cronograma	Atividades mais visíveis em gerência de projeto, que consistem em garantir que as atividades do projeto ocorram dentro dos tempos previamente definidos.
Custos	Atividades que buscam garantir que o projeto ocorra dentro do orçamento definido.
Qualidade	Do ponto de vista externo, visa garantir que o produto atenda às expectativas do cliente; do ponto de vista interno, visa garantir que o produto seja suficientemente maleável para não dificultar desnecessariamente o trabalho da equipe.
Recursos	Atividades de aquisição, dispensa, formação e motivação da equipe, bem como de alocação de funções e relações hierárquicas.
Comunicações	Controle das comunicações internas e externas ao projeto.
Riscos	Uma das áreas mais importantes da gerência de projetos, implica acompanhar o nível de probabilidade e impacto dos riscos e tomar medidas para diminuí-los.
Aquisições	Atividades relacionadas à aquisição de produtos ou serviços necessários ao projeto que não sejam produzidos ou fornecidos pela equipe de desenvolvimento.
Partes interessadas	A satisfação dos interessados frequentemente define o sucesso do projeto. Assim, o projeto deve manter boa comunicação com os interessados de forma a garantir que suas necessidades sejam satisfeitas.

A estrutura de processos do PMBOK é apresentada numa matriz, conforme mostrado na Tabela 9.2. Nesta matriz, as linhas são as áreas de gerenciamento e as colunas correspondem aos grupos de processo ou fases. Assim, o gerente pode se guiar, conforme a fase do projeto, identificando todos os processos que devem ser realizados para atender às diferentes áreas de gerenciamento.

Tabela 9.2 Processos do PMBOK identificados de acordo com a área de gerenciamento e grupo de processo (Fonte: PMI, 2017)

Áreas de gerenciamento	**Grupos de processo**				
	Iniciação	**Planejamento**	**Execução**	**Monitoramento e controle**	**Encerramento**
Integração	•Desenvolver o termo de abertura do projeto	•Desenvolver o plano de gerenciamento do projeto	•Orientar e gerenciar o trabalho do projeto •Gerenciar o conhecimento do projeto	•Monitorar e controlar o trabalho do projeto •Realizar o controle integrado de mudanças	•Encerrar o projeto ou fase
Escopo		•Planejar o Gerenciamento do Escopo •Coletar os requisitos •Definir o escopo •Criar a EAP (WBS)		•Validar o escopo •Controlar o escopo	
Cronograma		•Planejar o gerenciamento do Cronograma •Definir as atividades •Sequenciar atividades •Estimar os recursos das atividades •Estimar as durações das atividades •Desenvolver o cronograma		•Controlar o cronograma	
Custos		•Planejar o gerenciamento dos Custos •Estimar custos •Determinar o orçamento		•Controlar os custos	
Qualidade		•Planejar o gerenciamento da qualidade	•Gerenciar a qualidade	•Controlar a qualidade	
Recursos		•Planejar o gerenciamento de recursos	•Mobilizar recursos •Desenvolver a equipe •Gerenciar a equipe	•Controlar recursos	
Comunicações		•Planejar o gerenciamento das comunicações	•Gerenciar as comunicações	•Monitorar as comunicações	
Riscos		•Planejar o gerenciamento dos	•Implementar respostas aos riscos	•Monitorar os riscos	

Wazlawick, R. S. *Engenharia de Software: Conceitos e práticas*. 2ª. ed. Elsevier, 2019.

		riscos •Identificar os riscos •Realizar a análise qualitativa dos riscos •Realizar a análise quantitativa dos riscos •Planejar as respostas aos riscos			
Aquisições		•Planejar o gerenciamento das aquisições	•Conduzir as aquisições	•Controlar as aquisições	
Partes interessadas	•Identificar as partes interessadas	•Planejar o engajamento das partes interessadas	•Gerenciar o envolvimento das partes interessadas	•Controlar o envolvimento das partes interessadas	

Cada um destes 47 processos apresentados na Tabela 9.2 é detalhadamente descrito pelo PMBOK em termos de entradas e saídas, compostas por artefatos e ferramentas e técnicas que se aplicam às entradas para produzir as saídas. Essa organização das atividades mostra como é complexo e multidimensional o trabalho do planejador e do gerente de projeto.

A partir desta sexta edição (PMI, 2017) o PMBOK passou também a tecer considerações para ambientes ágeis, iterativos e adaptativos, aproximando, assim, mais o guia da realidade de muitas empresas que desenvolvem software e adotam modelos ágeis.

9.3 PRINCE2 – *Projects IN Controlled Environments* 2

Projects IN Controlled Environments 2 (Projetos em ambientes controlados 2), ou simplesmente *PRINCE2*, é um método estruturado de gerência de projetos bastante utilizado em várias partes do mundo. Prince2 foi lançado em 1996, como sucessor de outros métodos mais antigos. Desde 2006, o método tem sido revisado e atualizado, além de estar se tornando mais leve e compatível com Scrum. A versão atual é conhecida como "PRINCE2 2017" (AXELOS, 2017). Desde 2013 o modelo é propriedade da *AXELOS Limited*, mas continua sendo disponibilizado gratuitamente.

Além do manual de melhores práticas PRINCE2 que vem sendo desenvolvido há vários anos, a versão atual conta também com um guia ágil identificado como *PRINCE2 Agile*, lançado em 2015, que combina as melhores práticas de PRINCE2 com a cultura ágil.

Originalmente ele foi criado para a indústria de TI, mas atualmente pode ser aplicado em praticamente qualquer área, pois é um framework que procura apresentar as práticas de forma independente do tipo de produto, ou sejam isolando os aspectos ligados ao projeto (cronograma, custo, escopo etc.) dos aspectos ligados ao produto (software, construção civil, produto alimentício etc.).

O modelo se baseia em sete princípios que são apresentados na Tabela 9.3.

Tabela 9.3 Princípios de PRINCE2

Princípio	**Explicação**
Justificação continuada de negócio	Existe uma justificativa para iniciar o projeto e ela deve continuar válida ao longo dele, tendo sido documentada e aprovada. Em PRINCE2, a justificativa é documentada no *caso de negócio*, que dirige o processo de tomada de

	decisão e garante que o projeto permaneça alinhado com os objetivos e benefícios de negócio.
Aprender com a experiência	Espera-se que as equipes e o projeto aprendam com a experiência. Lições aprendidas são identificadas, registradas e praticadas ao longo do ciclo de vida do projeto. Em PRINCE2, as lições aprendidas são registradas no *lessons log*, ou diário de lições, e se tornam parte do relatório de lições aprendidas preparado pelo gerente de projeto ao final de cada estágio e ao final do projeto.
Papéis e responsabilidades definidos	Todos os projetos têm papéis e responsabilidades definidos e acordados, engajando todos os aspectos das organizações envolvidas interna e externamente. Os papéis devem ser definidos de forma que cada participante saiba exatamente o que se espera dele.
Gerenciar por estágios	O gerenciamento de projetos em PRINCE2 é compatível com os modelos baseados em iterações. Ao final de cada iteração, o projeto é revisado para verificar se atingiu os objetivos da iteração e se vai produzir a entrega prevista no caso de negócio. Isso é feito pelo uso de dois níveis de planejamento: longo prazo e curto prazo, sendo o segundo bem mais detalhado do que o primeiro.
Gerenciar por exceção	Os projetos PRINCE2 definem limites de tolerância para tempo, custo, qualidade, escopo e risco. Esses limites são usados para definir os níveis de autoridade delegada. Isso permite que a gerência seja feita dentro de um processo de gerência por exceção, ou seja, se esses limites de tolerância forem excedidos ou se for previsto que eles serão excedidos, então um nível superior de gerência deve ser acionado para decidir como proceder.
Foco nos produtos	PRINCE2 foca na definição e entrega de produtos que satisfazem os critérios de qualidade estabelecidos. Isso inclui o produto final de um projeto, bem como outros subprodutos significativos gerados ao longo do ciclo de vida do projeto. Essa abordagem orientada ao produto resulta na definição e geração de produtos sobre os quais se obtém concordância.
Personalização para se ajustar ao ambiente de trabalho	Prince2 deve ser personalizado para se adequar ao ambiente de trabalho, ou seja, tamanho, complexidade, importância, capacidades e riscos do projeto. Porém, quando se está personalizando o método, é importante não omitir nenhuma parte, pois todas elas são interligadas.

Além disso, o método também possui sete temas que descrevem aspectos críticos do gerenciamento de sistemas e devem ser considerados pelo gerente ao longo do ciclo de vida do projeto. Estes temas são apresentados na Tabela 9.4.

Tabela 9.4 Temas (aspectos críticos) em PRINCE2

Tema	**Explicação**
Caso de negócio	O caso de negócio dirige toda a tomada de decisões ao longo do projeto. Ele é criado no início do projeto e deve justificar os investimentos inicial e continuado. Permite que o gerente de equipe defina, a qualquer momento, se o projeto é viável, desejável e factível. O caso de negócio é mantido atualizado ao longo do projeto com relação a riscos, benefícios e custos.

Organização	A organização estabelece os papéis e as responsabilidades ligados ao projeto. Os projetos PRINCE2 são organizados em quatro níveis de decisão: *corporação* ou gerenciamento de programa, responsável pela encomenda do projeto; *diretoria*, que fornece direção e governança; *gerência de projeto*, que lida com as questões do dia a dia do projeto; e *gerência de equipe*, que se ocupa com a entrega dos produtos do projeto.
Qualidade	Procura garantir que o produto atenda ao seu propósito. A abordagem definida na estratégia de gerenciamento de qualidade requer que exista um entendimento explícito sobre o escopo do projeto, bem como critérios de qualidade contra os quais o produto será avaliado. Em outras palavras, o foco da qualidade está em fazer o produto atender aos requisitos (especialmente aos de qualidade). O método possui duas formas de aplicação de controle de qualidade: *in process*, que implica construir os produtos com qualidade ao longo do projeto, e *appraisal*, usado para verificar se os produtos acabados satisfazem os critérios de qualidade.
Planos	Planos são usados para definir como, quando e por quem os produtos serão gerados. Planos são usados para permitir e facilitar a comunicação efetiva e o controle. Eles devem estar alinhados com o caso de negócio e precisam ser aprovados por todos os níveis de gerenciamento. Existem três níveis de planos do PRINCE2: plano de projeto, plano de estágio e, opcionalmente, plano de equipe. Adicionalmente, planos de exceção poderão substituir um dos outros planos em situações em que os níveis de tolerância de gerenciamento forem excedidos. Prince2 requer que o planejamento seja orientado ao produto, estabelecendo que primeiramente o produto do trabalho deve ser decidido e somente depois as atividades, dependências e recursos podem ser determinados, sempre com o objetivo de produzir o produto especificado.
Riscos	Para PRINCE2, os riscos são incertezas que podem ser tanto positivas (oportunidades) quanto negativas (ameaças). O gerenciamento de risco trata da identificação proativa, da avaliação e do controle dos riscos do projeto de forma a maximizar as chances de sucesso. A estratégia de gerenciamento de riscos, definida durante a iniciação do projeto, possui cinco passos: identificar a causa, evento e efeitos do risco; analisar sua probabilidade e impacto; planejar as respostas necessárias; implementar as respostas como requerido; e comunicar os riscos interna e externamente.
Mudança	Envolve a gerência de configuração, problemas e solicitações de mudança. Mais detalhes sobre essas disciplinas serão vistos no Capítulo 10.
Progresso	Refere-se aos mecanismos usados para monitorar e comparar o atual estado do projeto em relação aos planos. Esse mecanismo também permite que o gerente de projeto faça previsões de performance baseadas nas tendências atuais e, assim, realize ações proativas visando a alternativas de projeto quando necessário.

Os sete princípios e sete temas são então amalgamados em sete processos, conforme mostrado na Tabela 9.5.

Tabela 9.5 Processos em PRINCE2

Processo	**Atividades**
Dar partida em um	Indicar um executivo e um gerente de projeto, planejar e indicar uma equipe de gerenciamento de projeto, preparar um resumo executivo do projeto, definir a

projeto	abordagem do projeto e planejar para o próximo estágio.
Iniciar um projeto	Planejar qualidade, planejar o projeto, refinar o caso de negócios e riscos, definir os meios de controle do projeto, definir os arquivos do projeto e montar o documento de início de projeto.
Dirigir um projeto	Autorizar o início do projeto, autorizar um estágio ou plano de exceção, dar orientações *ad hoc* e confirmar o fechamento do projeto.
Controlar um estágio	Autorizar um pacote de trabalho, avaliar o progresso, capturar e examinar assuntos do projeto, revisar o status do estágio, relatar destaques, tomar ação corretiva e receber um pacote de trabalho completo.
Gerenciar limites de estágios	Planejar um estágio, atualizar um plano de projeto, um caso de negócio e a lista de riscos, relatar o final de um estágio e produzir um plano de exceção.
Gerenciar entrega de produto	Aceitar um pacote de trabalho, executá-lo e entregá-lo.
Fechar um projeto	Encerrar (*decommissioning*) o projeto, identificar ações posteriores e revisar a avaliação do projeto.

O método PRINCE2 estipula três níveis de certificação: *foundation*, para os que aprenderam os fundamentos do método; *practicioneer*, para aqueles que vão gerenciar projetos usando o método; e *certification*, para aqueles que buscam ainda mais competência em gerenciamento de projetos.

9.4 Condução de Projeto de Software

Muitas vezes, após o planejamento, um projeto de software precisa ser executado durante um longo período de tempo. Assim, é papel do gerente garantir que as variáveis de tempo, recursos, qualidade e escopo sejam mantidas nos valores esperados.

Após o planejamento, espera-se que riscos já tenham sido avaliados e existam planos para mitigá-los, responsáveis nomeados, recursos alocados, e que um processo de desenvolvimento já esteja definido. Resta executar o projeto – e pode ser que nessa hora tudo comece a dar errado.

Um projeto, mesmo bem planejado, pode falhar por vários motivos: erros da equipe, erros no próprio projeto, erros na concepção do processo ou, ainda, fatores imprevistos. Isso decorre principalmente do fato de que os executores do projeto são pessoas, e não máquinas, e de que um projeto não é um programa de computador que roda de forma previsível. Muitos fatores de incerteza estão envolvidos mesmo nos projetos mais bem planejados e gerenciados.

Staa (2003) indica que um dos maiores desafios que o gerente de projeto enfrenta no momento do acompanhamento da execução de um projeto é a *indisciplina*. Membros da equipe podem não seguir os padrões, prioridades ou os prazos estabelecidos. O folclore da Ciência da Computação apresenta os "gênios" como superdesenvolvedores que não seguem regras, não têm horários preestabelecidos, são desorganizados e, muitas vezes, têm problemas de higiene pessoal. Se não houver disciplina no ambiente de trabalho, esses "heróis" poderão ressurgir e tomar o projeto como refém.

Não se devem confundir os métodos ágeis, mesmo os mais radicais, como o XP, com indisciplina e caos. Para que um modelo ágil funcione na prática, a equipe deve ser muito bem disciplinada. No caso de *Scrum*, a equipe deve ser necessariamente autodisciplinada, pois ela se autogerencia.

Assim, o uso de métodos ágeis não é desculpa para fazer as coisas de qualquer maneira. Mesmo

que os desenvolvedores trabalhem com criatividade, buscando soluções inovadoras e às vezes ousadas para os problemas, devem entender que existem as prioridades de projeto, que ou são discutidas e mudadas, ou são mantidas e seguidas.

Além disso, existem padrões a serem seguidos. O desenvolvedor baixa uma versão do componente no sistema de controle de versões e tem total liberdade para trabalhar em sua cópia. Mas, no momento de salvar uma nova versão no sistema, o que ele colocar ali vai afetar o restante da equipe. Então, essa versão precisa estar de acordo com os padrões estabelecidos, testada e estabilizada.

É responsabilidade do gerente de projeto, portanto, identificar se há algum tipo de desvio nocivo e motivar os desenvolvedores, conscientizando-os da necessidade de trabalhar em harmonia (lembrando que *harmonia* não significa cada um fazer o que quer, mas todos seguirem as regras que o grupo estabeleceu).

Staa (2003) identifica três perfis de gerência em relação à equipe:

- *Ditador*: é o mais danoso dos três. Faz todo o planejamento e as estimativas, e determina sozinho quem faz o quê e quando. Gera belíssimos diagramas Gantt que ninguém leva a sério e usa de força e ameaça para fazer projetos que não estão indo bem voltarem para os trilhos, o que nem sempre consegue.
- *Coordenador*: ouve os outros e faz as previsões e o planejamento em conjunto com a equipe. Faz revisões do planejamento periodicamente. Seus diagramas Gantt são levados mais a sério, pois representam um compromisso realista da equipe, e não apenas uma determinação do gerente. Seus projetos têm maior chance de terminar no prazo. Esse tipo de gestão costuma ser mais típico com métodos prescritivos.
- *Facilitador*: apenas agiliza e facilita o trabalho da equipe, mas não toma as decisões. A própria equipe define os prazos e o planejamento. Esse tipo de gerência costuma ser mais típico com os métodos ágeis.

Staa (2003) também apresenta os três objetivos de acompanhamento de projetos do CMMI, e acrescenta um quarto:

- Acompanhar os resultados e desempenhos reais, confrontando-os com o plano de desenvolvimento de software.
- Realizar ações corretivas e gerenciá-las até sua conclusão, sempre que resultados ou desempenhos reais se desviarem significativamente do que foi estabelecido (estimado) no plano de desenvolvimento de software.
- Assegurar que as alterações nos compromissos de software se deem através de acordo entre as pessoas e os grupos envolvidos.
- Acompanhar processos e metaprocessos, obtendo indicadores quanto à sua eficácia em instanciar planos e processos.

Os dois primeiros objetivos mencionados estão relacionados com o dia a dia do gerente de projeto e correspondem às atividades de verificação e controle por ações corretivas. O terceiro objetivo está relacionado às mudanças eventualmente necessárias no plano do projeto. O quarto relaciona-se com a necessidade de mudanças no processo quando se detecta que ele não é suficientemente adequado.

Finalmente, convém mencionar que um plano mal feito será difícil de se gerenciar. Assim, é fundamental que no momento do planejamento de um projeto as técnicas apresentadas no Capítulo 6 sejam observadas. Por exemplo, se o plano não prevê artefatos prontos em determinadas datas, fica difícil o gerente avaliar se as atividades previstas foram efetivamente realizadas. Como a verificação do gerente tende a ser pontual (ele avalia o *status* do projeto ou o de cada

desenvolvedor em determinados momentos), ele só será capaz de avaliar o estado e a qualidade de artefatos produzidos, mas não diretamente o estado e a qualidade das atividades em si.

9.5 Medição em Engenharia de Software

Um processo de gerência, para ser mais eficaz, precisa se basear em medições. Como saber se a tarefa não está sendo feita se não houver uma medida para chegar a essa conclusão? Como saber que a qualidade é inaceitável? De uma maneira ou de outra, o gerente de projeto vai acabar se envolvendo com a atividade de medição de software, na qual terá que aplicar uma ou mais métricas.

Inicialmente, alguns termos serão definidos para evitar confusão:

- *Medida*: valor obtido para alguma dimensão do software.
- *Métrica*: escala na qual os valores de uma medida são tomados.
- *Medição*: processo de obtenção de medidas.

O *Software Engineering Institute* (Mills, 1988) indica que uma boa métrica precisa ter cinco qualidades:

- Ser *simples*, ou seja, ter uma definição curta e fácil de ser compreendida.
- Ser o mais *objetiva* possível, isto é, não depender de opiniões. Se duas pessoas avaliarem o mesmo produto usando essa métrica, deverão obter o mesmo resultado.
- Ser *facilmente obtida*, isto é, ter custo de medição razoavelmente baixo.
- Ser *válida*, isto é, indicar um valor que seja útil e efetivamente representativo da grandeza que se pretende medir.
- Ser *robusta*, isto é, pequenas mudanças no produto devem gerar mudanças proporcionais na medida. Apenas grandes mudanças no produto podem produzir grandes mudanças na medida.

Apesar disso, nem sempre as métricas apresentam todas essas qualidades. Um bom exemplo de métrica é a contagem de linhas de código (KSLOC; Seção 7.3). Desde que os padrões de contagem tenham sido estabelecidos (o que efetivamente conta e o que não conta), essa métrica deve produzir sempre o mesmo resultado. Já a contagem de pontos de história (Seção 7.2) poderá apresentar variações significativas, dependendo da interpretação do desenvolvedor sobre a dificuldade para desenvolver a história de usuário. O método CII (Seção 7.4) tenta reduzir a subjetividade nos multiplicadores de esforço e fatores de escala ao estabelecer uma série de padrões para atribuição de notas. Ainda assim, a métrica tem uma carga de subjetividade significativa.

A Seção 11.4 apresenta mais alguns detalhes sobre medição e métricas referentes à qualidade de software e as subseções seguintes discutem classificações de métricas e o planejamento de um programa de métricas.

9.5.1 Classificações de Métricas

Existem várias classificações para métricas. Inicialmente, pode-se falar em métricas diretas, ou seja, aquelas que podem ser definidas e contadas de modo direto, sem necessidade de interpretação ou incerteza, como:

- Custo financeiro.
- Esforço em desenvolvedor-mês.
- Linhas de código (SLOC).
- Velocidade de execução em segundos.
- Memória em megabytes.
- Número de defeitos localizados (total ou relativo ao número de KSLOC).

- Complexidade ciclomática (Seção 13.4.1).

Outras métricas são indiretas e só podem ser determinadas a partir de uma definição operacional (Wazlawick, 2014), ou seja, um procedimento de medição que algumas vezes é *ad hoc*, e nem sempre será consenso. Exemplos de métricas indiretas são:

- Funcionalidade.
- Qualidade.
- Complexidade.
- Eficácia.
- Confiabilidade.
- Manutenibilidade.
- Usabilidade.

Nota-se que as métricas mais importantes estão fortemente ligadas aos aspectos de qualidade do software (Seção 11.1).

Um aspecto que sempre deve ser lembrado é que só vale a pena coletar medições quando se tem em mente um propósito específico, pois a coleta de dados poderá acarretar trabalho extra antes, ao longo e depois do projeto de desenvolvimento. Assim, a coleta de medições deve ser encarada como um investimento de tempo e esforço com o objetivo de melhorar algum aspecto do produto ou do processo de desenvolvimento.

Do ponto de vista dos processos de gerência, pode ser interessante agrupar as métricas em termos de sua utilidade para o gerente. Assim, temos:

- *Métricas de produtividade*: custo, pontos de função, pontos de história, pontos de caso de uso ou linhas de código, que podem ser usadas para verificar o andamento do projeto e possíveis desvios.
- *Métricas de qualidade*: número de defeitos, eficiência, confiabilidade e capacidade de manutenção. São usadas para avaliar se o produto satisfaz aos critérios de aceitação para uso por parte do cliente e também critérios internos, que afetam a eficiência da equipe.
- *Métricas técnicas*: outros aspectos ligados ao produto, não necessariamente à qualidade ou à produtividade, mas a aspectos inerentes do sistema, como complexidade ciclomática, modularidade, paralelismo, distribuição etc.

As métricas podem ser *absolutas* ou *relativas*. Por exemplo, um sistema com cinco erros não é necessariamente pior do que um sistema com dois erros. Se o primeiro sistema tiver um milhão de linhas e o segundo tiver cinco mil linhas, então o segundo terá mais erros por linha do que o primeiro.

Assim, a medida relativa é frequentemente usada, em especial quando se deseja avaliar a qualidade do produto e do trabalho. Há pelo menos quatro formas relevantes de relativizar uma métrica:

- *Pelo tamanho*: divide-se o valor absoluto da métrica pelo número de linhas de código.
- *Pela funcionalidade*: divide-se o valor absoluto da métrica pelo número de pontos de função, pontos de caso de uso ou pontos de histórias.
- *Pelo tempo*: divide-se o valor absoluto pelo período de tempo. Por exemplo, número de defeitos detectados por mês.
- *Por esforço*: divide-se o valor absoluto pelo esforço despendido, geralmente em desenvolvedor-mês ou desenvolvedor-hora. Por exemplo, o número de linhas de código produzidas por desenvolvedor-mês.

Talvez o trabalho mais formal na área de métricas esteja relacionado à medição da complexidade

do software, o que é útil para a área de testes. Outro tipo de métrica que recebeu muita atenção são as métricas de qualidade ligadas ao produto (Capítulo 11), embora às vezes haja competição entre elas. Por exemplo, aumentar a flexibilidade (desejável) pode diminuir a eficiência de tempo (indesejável).

Entretanto, algumas métricas de qualidade podem ser especialmente úteis ao gerente de projeto, pois melhorar suas medições provavelmente será muito salutar:

- *Métricas de defeitos*: o número de defeitos em um produto de software deveria ser uma métrica objetivamente contável. Porém, encontrar defeitos não é uma tarefa trivial. Assim, normalmente, a medição de defeitos é feita em função dos erros observados. A Seção 13.1.1 explica a diferença conceitual entre *erro* e *defeito*.
- *Métricas de confiabilidade*: uma vez que se tenha uma medida nominal dos defeitos encontrados em um produto de software, sua confiabilidade pode ser calculada para determinado período de tempo. Se um sistema complexo apresenta certa taxa de falhas ao longo de um mês ou de um ano, pelas Leis de Lehman (Seção 14.1) pode-se esperar que continue exibindo esse comportamento ao longo dos meses ou anos seguintes. Essas medidas podem ser tomadas tanto durante o processo de desenvolvimento quanto durante o período de operação pós-desenvolvimento do sistema.
- *Métricas de manutenibilidade*: embora a manutenibilidade seja, a princípio, uma métrica subjetiva, existem estudos que mostram que a complexidade do produto (a complexidade ciclomática, por exemplo) afeta o esforço necessário para encontrar e reparar defeitos no software. Assim, existe uma relação direta entre o número de defeitos encontrados e a complexidade do software com o esforço necessário para fazer manutenções. Novamente, essas atividades de manutenção podem ser tanto aquelas que ocorrem durante a operação do software quanto as modificações que ocorrem durante o desenvolvimento.

Deve-se ficar atento, porém, para o fato de que essas métricas precisam ser sempre interpretadas em seu contexto. Por exemplo, a detecção de muitos defeitos no software pode significar tanto que a atividade de teste está sendo bem conduzida quanto que as atividades de programação estão sendo mal conduzidas. Mas defeitos detectados por usuários durante o uso do sistema costumam ser péssimas notícias.

9.5.2 Planejamento de um Programa de Métricas

Para que um gerente possa utilizar métricas para dirigir suas atividades, ele deve primeiro implantar um *programa de métricas*. Segundo o SEI (Mills, 1988), essa atividade deve ser muito bem planejada.

Inicialmente, devem-se estabelecer os objetivos do programa de métricas: Quais falhas ele vai corrigir? Quais aspectos ele vai tentar melhorar? Em seguida, devem ser estimados os custos do programa e deve ser obtido o apoio da gerência superior, pois esse tipo de ação normalmente precisa de um suporte razoável para ocorrer. Em relação aos custos, estes podem ser divididos em custos iniciais de implantação e custos de manutenção do programa.

Em função dos objetivos e do aporte financeiro inicialmente comprometido, deve-se escolher o conjunto de métricas a serem implementadas e o modelo de avaliação (por exemplo, para estimar o tamanho de um projeto, podem-se usar os modelos CII, Pontos de Função ou Pontos de Caso de Uso, cada qual com suas peculiaridades e custos específicos). Durante o processo de seleção das métricas e modelos devem-se observar os seguintes pontos:

- *Habilidade projetada para satisfazer objetivos*: as métricas e modelos disponíveis devem ser analisados e comparados em relação a sua capacidade de satisfazer os objetivos do programa

de métricas.

- *Dados e custos necessários estimados*: os modelos que são capazes de satisfazer os objetivos devem ser comparados em função de seu custo de implantação e manutenção e da quantidade e variedade de dados necessários para funcionar. Normalmente, os modelos mais econômicos são preferíveis.

Uma vez que o modelo tenha sido escolhido, deve-se identificar e refinar os dados que serão coletados. Apenas dados que possam satisfazer a algum objetivo imediato ou de longo prazo devem ser coletados. Para executar essa atividade, deve-se observar o seguinte:

- *Especificidade dos dados*: os dados devem ser definidos e obtidos ao longo de todo o ciclo de desenvolvimento do software. Preferencialmente, deve-se identificar *quando* os dados foram obtidos. Isso permite analisar diferentes significados em diferentes fases ou para diferentes atividades do processo de desenvolvimento.
- *Procedimentos de obtenção de dados*: uma vez que os dados específicos tenham sido definidos, os procedimentos para sua coleta e as pessoas responsáveis devem ser identificados.
- *Manutenção do banco de dados*: uma vez que o banco de dados de medições passará a ser um importante patrimônio da empresa, os recursos para sua perfeita manutenibilidade devem ser definidos e destinados.
- *Previsões refinadas de esforço e custo*: com as informações obtidas nos itens anteriores, deve ser possível obter uma estimativa bem mais realista de custo e esforço para a implantação do programa de métricas.

Assumindo que todos os passos anteriores foram executados com sucesso e os custos são aceitáveis, o programa de métricas pode ser iniciado. Os seguintes itens ainda precisam ser enfatizados nessa fase:

- *Esclarecimento de uso*: os objetivos do programa de métricas devem ficar claros desde o início. Mas, no momento de iniciar seu uso, pode ser importante relembrá-los a toda a equipe. As pessoas devem ser informadas sobre as medidas que serão obtidas e o uso que será feito delas. Muito cuidado deve ser tomado especialmente se as medidas forem utilizadas para avaliação de membros da equipe. Sem uma ampla discussão e aceitação das métricas, essas iniciativas poderão dar origem a estresse e até sabotagens ao programa.
- *Pessoal responsável*: os responsáveis pela coleta, manutenção e interpretação dos dados devem ser definidos e informados. Deve-se lembrar que essa atividade deve ser executada continuamente ao longo de qualquer projeto, pois muitos dados poderão não ser mais obtidos depois que ele terminar.

Para que o programa de métricas tenha sucesso, ele deve ser continuamente usado, avaliado e reajustado. Os seguintes aspectos são relevantes:

- *Avaliação de resultados*: os resultados devem ser cuidadosamente resumidos e comparados com a realidade subjetivamente observada. Por vezes, algum desvio em relação à percepção da equipe pode ser resultado de erros no processo de obtenção dos dados.
- *Ajuste do modelo*: muitos modelos de medição exigem que determinadas constantes sejam calibradas ao longo de seu uso. Assim, esses procedimentos de calibração devem ser executados sempre que o modelo assim o exigir.

Finalmente, convém lembrar que, em termos de capacidade de processos e maturidade de empresas, é absolutamente necessário que exista um plano de métricas e que ele seja efetivamente usado para melhorar aspectos dos processos da empresa. Os níveis mais altos de maturidade dos modelos de qualidade só são atingidos quando existe um plano de métricas que seja efetivamente colocado

em prática.

9.6 Revisão e Avaliação

Nos métodos ágeis, como *Scrum*, a revisão e a avaliação de projeto são previstas e ocorrem informalmente nas reuniões diárias em pé e com mais formalidade nas reuniões de fechamento de iteração.

Outros métodos poderão definir reuniões de revisão de projeto, de forma periódica, seja ao final de uma iteração ou fase, seja a qualquer momento.

É importante que, para serem efetivas, as reuniões sejam planejadas com antecedência. As pessoas envolvidas devem ser comunicadas, o local adequado deve ser reservado e, o que é mais importante, os objetivos da reunião devem ser claramente transmitidos. Quando se fala em objetivos, deve-se entender isso no sentido de definir quais serão os *artefatos de saída* da reunião. Pode-se ver, então, que a reunião de revisão e avaliação também é vista como uma atividade de projeto. Quando os participantes de uma reunião começam a divagar e a discutir sem objetividade, cabe ao condutor da reunião interromper a divagação e solicitar sugestões de encaminhamento para os assuntos pendentes, de forma que os objetivos da reunião possam ser atingidos o mais rapidamente possível.

Além disso, sempre que possível, todo o material a ser usado na reunião deve ser distribuído previamente. A reunião não é o momento de apresentar novidades ou surpresas para a equipe. É recomendável que, sempre que possível, os assuntos sejam adiantados; caso contrário, a reunião poderá se estender demais.

Não é recomendado que, na reunião de revisão e avaliação, se faça com o grupo uma análise minuciosa, por exemplo, de todo o código produzido pela equipe. Isso é impossível por questões de tempo. As revisões e os relatórios já deverão estar prontos para a reunião, tendo sido feitos por aqueles a quem a tarefa foi delegada. No momento da reunião, apenas uma visão geral deve ser apresentada e, se necessário, o aprofundamento de um ou outro pontos críticos.

Para a pauta da reunião sugere-se que o gerente de projeto procure apresentar os seguintes aspectos:

- Os principais marcos de projeto (*milestones*) alcançados.
- Os desvios que eventualmente tenham ocorrido em relação ao plano de desenvolvimento do projeto.
- Modificações significativas na alocação de esforço, ou seja, se mais (ou menos) tempo do que o previsto foi dedicado a alguma atividade.
- Modificações significativas em termos de despesas, ou seja, se as atividades consumiram mais (ou menos) itens de orçamento do que o previsto.
- Mudanças no escopo estimado de trabalho, ou seja, se houve alguma modificação em termos das funcionalidades ou outros aspectos que inicialmente consistiam em um objetivo da fase ou ciclo e que foram mudados, removidos ou acrescentados.
- Mudanças na métrica de qualidade, ou seja, se por alguma razão a forma de medir a qualidade do trabalho foi mudada.
- *Status* dos riscos de projeto, ou seja, quais riscos continuam sendo muito importantes e quais tiveram sua importância alterada ao longo da fase ou do ciclo, especialmente caso a alteração tenha sido no sentido de um risco ficar mais importante do que era anteriormente.
- Novos riscos identificados.
- Problemas relevantes que tenham surgido e ainda não tenham sido resolvidos.

- Andamento de eventuais ações que tenham sido determinadas em reuniões anteriores.
- Lições aprendidas para projetos futuros.

Durante a reunião e especialmente ao final dela devem ser tomadas decisões sobre ações a serem executadas, sempre que necessário. O gerente deve indicar responsáveis e prazos para as ações mais urgentes. Já as ações não urgentes devem ser colocadas na lista de modificações ou na lista de riscos de projeto para serem tratadas oportunamente, de acordo com sua prioridade.

9.7 Fechamento

O fechamento ou encerramento de um projeto (ou de uma fase de um projeto) tem como objetivo formalizar a entrega do produto e a sua aprovação pelo cliente.

Será uma reunião ainda mais formal do que as reuniões periódicas de avaliação do projeto e, possivelmente, deverá envolver um número maior de interessados. Dentre os itens tratados nas reuniões de fechamento de projeto, pode-se imaginar que vários serão semelhantes aos itens abordados nas reuniões de avaliação. Porém, em uma avaliação final, será importante observar os seguintes aspectos:

- Se todos os objetivos iniciais do projeto foram alcançados.
- Se houve algum desvio importante ao longo do projeto em relação ao seu plano.
- Se houve alguma mudança significativa em relação ao esforço previsto e executado (o que poderá servir de entrada para a calibragem de métricas de esforço).
- Se houve alguma mudança significativa em relação aos recursos alocados e consumidos.
- Se houve mudanças de escopo importantes.
- Os resultados finais das métricas de qualidade e, eventualmente, sua evolução.
- Riscos de operação, caso tenham sido identificados.
- Problemas relevantes que tenham surgido e ainda não tenham sido resolvidos.
- Lições aprendidas para projetos futuros.

Além de formalizar a entrega do produto, a reunião de encerramento do projeto também será uma oportunidade importante para que sejam produzidos documentos reportando o desempenho da organização, bem como os problemas de percurso e as soluções encontradas. Dessa forma, a inteligência criada durante o projeto fica registrada para possível uso futuro.

A finalização do projeto também é importante para que fique claro para todos os envolvidos que as atividades de desenvolvimento terminaram e que quaisquer novas modificações serão alvo de projetos futuros (ou do processo contínuo de manutenção).

REMISSIVO DO CAPÍTULO

Wazlawick, R. S. *Engenharia de Software: Conceitos e práticas*. 2ª. ed. Elsevier, 2019.