Vinícius Souza

Definindo o processo de software

COLEÇÃO EAD
EDITORA UNISINOS

DEFININDO O PROCESSO DE SOFTWARE

VINÍCIUS COSTA DE SOUZA

EDITORA UNISINOS
2011

APRESENTAÇÃO

Todas as empresas são formadas por um conjunto de processos, os quais são executados diariamente para que os objetivos do negócio sejam atingidos. Tais processos precisam ser cuidadosamente executados e gerenciados para que os resultados sejam alcançados com a qualidade esperada. Uma das formas de garantir a boa execução dos processos organizacionais é a realização da definição desses processos, através da qual determina-se a melhor forma para a sua execução e gerenciamento, o que passa a se tornar um padrão organizacional.

No caso de software, que é desenvolvido em equipe, através de muitas atividades complexas e que está sob influência de vários fatores externos, o benefício da definição de processos é extremamente positivo. Segundo muitos autores da área de engenharia de software, a qualidade de um sistema está diretamente relacionada à qualidade do processo utilizado na sua construção.

Assim, este livro apresenta os principais conceitos relacionados ao mapeamento de processos organizacionais, bem como o processo de software e seus modelos. Para isso, o livro está organizando em oito capítulos. No primeiro capítulo é apresentado o conceito de processo, bem como seus componentes e características. Já no capítulo dois são apresentadas as técnicas e ferramentas para o mapeamento e a modelagem de processos organizacionais e, no capítulo três, as principais notações utilizadas no mapeamento de processos: *Unified Modeling Language* (UML) e *Business Process Management Notation* (BPMN). No capítulo quatro são apresentadas as importantes etapas de avaliação e mudança de processos. O capítulo cinco apresenta o processo de software como um todo, incluindo suas atividades fundamentais e de apoio. Já no capítulo seis são apresentas algumas das atuais ferramentas de apoio ao processo de software, as chamada ferramentas CASE (do inglês *Computer-Aided Software Engineering)*. Finalmente, nos capítulos sete e oito, são apresentados alguns modelos de processo de software e modelos de melhoria.

SUMÁRIO

CAPÍTULO 1

INTRODUÇÃO

Neste capítulo você vai conhecer o conceito de processo, bem como seus componentes e características. Além disso, vai compreender o que é um processo definido e a forma como podemos organizar os processos de uma empresa, através da sua hierarquização. Também vai conhecer os benefícios da gestão organizacional baseada em processos.

1.1 Processos organizacionais e seus componentes

As organizações são constituídas por uma complexa combinação de processos, os quais são executados através de seu capital humano, instalações e equipamentos para que os objetivos do negócio sejam atingidos. Assim, conforme os desempenhos dos processos, podem afetar positivamente ou negativamente a organização como um todo.

Um processo trata-se de um conjunto parcialmente ordenado de passos, constituídos por atividades, métodos, práticas e transformações, usado para atingir uma meta (BARBARA, 2006). Em outras palavras, um processo é um conjunto de atividades realizadas para se atingir um resultado.

Conforme ilustrado na figura 1, todo o processo é constituído por quatro componentes: entradas, condições, meios e saídas.

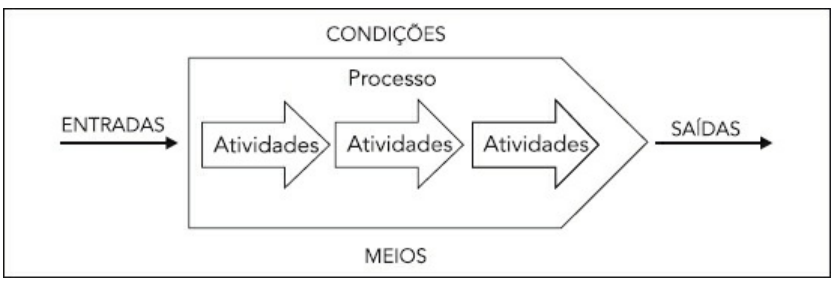

Figura 1 – Componentes de um processo.

As entradas são os insumos do processo, ou seja, tudo aquilo de que necessitamos para iniciar a execução do processo. Podem ser materiais, como a borracha para a produção de um pneu ou as rodas para a montagem de um carro; informações, como a quantidade de uma determinada substância para a produção de um medicamento; ou ainda eventos, os quais disparam o início do processo.

Os meios são as pessoas, ferramentas e instalações necessárias para a execução do processo. Por exemplo, para executarmos o processo de produção de um armário de madeira precisamos de um conjunto de entradas como a própria madeira, os pregos, as informações sobre as dimensões do armário, entre outras. Os meios, nesse exemplo, são representados pelo marceneiro, o martelo e demais ferramentas necessárias, além da marcenaria onde o trabalho será realizado.

Já as condições para a execução de um processo são definidas por regulações legais, regras definidas pelo responsável pelo processo, normas e políticas internas de cada empresa e exigências do contexto atual, no qual o processo está inserido. Por exemplo, existem várias condições para a execução do processo de criação de uma matéria jornalística como a regulação legal, que não permite que os jornalistas tenham uma carga horária de trabalho superior a seis horas, prazo de antecedência definido pelo jornal para a publicação das matérias na edição do dia seguinte, entre outras.

As saídas são os resultados do processo, os quais podem ser tangíveis como um carro ou intangíveis como um serviço prestado, que não gera algo concreto como resultado.

A figura 2 apresenta as diferentes possibilidades de entradas, condições, meios e saídas de um processo.

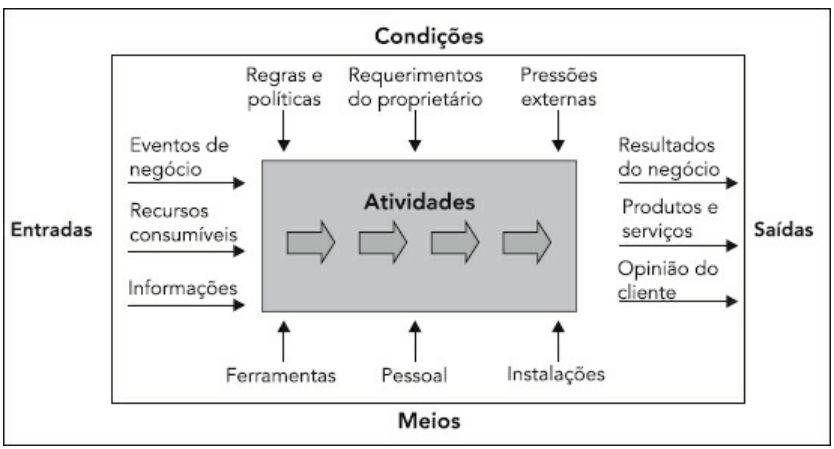

Figura 2 – Possibilidades de entradas, saídas, meios e condições de um processo.

Fonte: Autor.

1.2 Hierarquia de processos

Como as empresas são constituídas por muitos processos, é necessário organizarmos todos esses processos em categorias de forma a facilitar sua compreensão e gerenciamento. Uma das possibilidades para essa organização é a hierarquização dos processos (Figura 3), a qual divide os processos nas seguintes categorias: processos empresariais, processos, subprocessos, atividades e tarefas.

A hierarquização inicia com os processos mais complexos e críticos para o negócio e estende-se até os processos mais simples, os quais podem ser realizados rapidamente por apenas uma pessoa.

Os processos empresariais são os processos críticos para o andamento do negócio e para a satisfação do cliente. Erros nesses processos prejudicam a lucratividade e a imagem da empresa, pois são facilmente percebidos pelo cliente. Exemplos de processos dessa categoria são: o processo de matrícula de uma instituição de ensino e o processo de venda de uma loja. Em ambos os casos, se o processo for lento ou tiver qualquer tipo de problema, o cliente será imediatamente afetado, prejudicando a imagem da empresa.

Já os chamados processos são aqueles que iniciam e terminam com o cliente externo. Essa categoria é muito semelhante à anterior, diferenciando-se apenas pelo fato de que esses não são críticos para o negócio e não afetam diretamente a satisfação do cliente. São processos com um grau de criticidade um pouco menor do que os processos empresariais. Como exemplo, podemos citar a atualização de endereço que o cliente pode precisar fazer antes de efetuar sua matrícula ou realizar a compra de um produto em uma loja. A atualização de endereço não é o objetivo final do cliente, mas sim uma etapa necessária para que ele atinja o seu objetivo. Da mesma forma, a atualização de endereço não é o objetivo de negócio da empresa.

Os subprocessos são os processos que envolvem mais de um setor da empresa para serem executados. Neste caso, podemos citar como exemplo o subprocesso de análise de crédito, o qual inicia no setor de vendas e passa pelo setor financeiro para ser finalizado.

Classificamos como atividades aquelas que são realizadas em um único setor de empresa, podendo necessitar de diferentes pessoas desse mesmo setor para que seja concluída. Podemos citar como exemplo de atividade a aprovação do crédito, a qual necessita inicialmente ser realizada por um analista de crédito e, posteriormente, passar por um aval final do gerente financeiro.

Finalmente, as tarefas são aquelas realizadas por apenas uma pessoa, já que são simples e rápidas de serem executadas. Um exemplo de tarefa pode ser a assinatura da

aprovação de crédito, realizada pelo gerente financeiro.

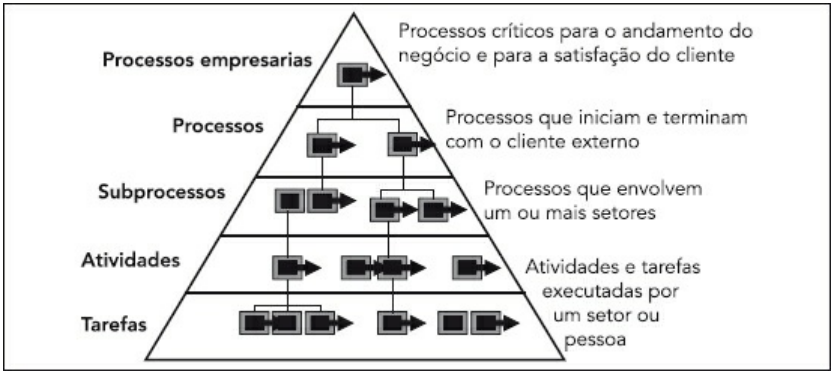

Figura 3 – Hierarquia dos processos.
Fonte: Autor.

Como um processo empresarial é constituído de vários processos, os quais possuem subprocessos que são formados por atividades e tarefas, a hierarquização de todos esses processos facilita sua compreensão e organização.

1.3 Características dos processos

Todos os processos, independente de sua classificação, possuem características em comum. É importante que essas características sejam conhecidas para que seja possível realizar um melhor controle e gerenciamento dos processos, a fim de que eles possam cumprir adequadamente seus objetivos. As características mais comuns dos processos são apresentadas no quadro 1.

Quadro 1 – Características comuns entre os processos

Gerenciabilidade	Todo o processo deve estar sob a responsabilidade de alguém.
Efetividade	As relações entre fornecedores e clientes devem ser claramente definidas. Os fornecedores são todos que entregam as entradas necessárias para a execução do processo. Já os clientes são aqueles que recebem as saídas de um processo. Cabe salientar que existem fornecedores e clientes tanto internos, dentro da própria empresa, como externos.
Transferibilidade	Todo o processo deve possuir documentação que permita transferir informações sobre o seu funcionamento.

Riscos	Todo o processo possui riscos, que são os possíveis problemas que podem ocorrer durante sua execução. Assim, precisamos conhecer quais são esses riscos e monitorar o processo para evitar que eles ocorram.
Mensurabilidade	Todo o processo deve possuir pontos de controle, que são medidas que fazemos sobre o processo para verificar sua eficácia.
Alterabilidade	Todo o processo deve possuir mecanismos que permitam verificar a necessidade de realizarmos alterações para o seu aperfeiçoamento.
Rastreabilidade	Devem existir formas de rastrear os processos, para permitir que as causas de possíveis problemas sejam identificadas e resolvidas.

1.4 Definição de processos

Dizemos que um processo está definido quando ele possui documentação que detalha, pelo menos, o que é feito, como é feito, quando, quem faz, o que usa e o que produz.

Essa documentação é o registro do conhecimento que as pessoas têm sobre o processo e, através dela, é possível padronizar a sua execução, garantindo-se que os resultados esperados sejam atingidos com maior confiabilidade.

Sabemos que toda empresa é formada por um conjunto de processos e, assim, esses processos precisam ser devidamente gerenciados ou as empresas precisam torcer para que eles deem certo. Dessa forma, a definição de processos, que resulta na sua documentação, é o primeiro e fundamental passo para a gestão por processos.

1.5 Gestão por processos

Para que se obtenha excelência e sucesso em um negócio, é necessário que todas as atividades sejam compreendidas e gerenciadas segundo uma visão por processos, conforme ilustrado na figura 4. Isso porque é muito importante que sejam conhecidos os clientes desses processos, seus requisitos e o que cada atividade agrega de valor na busca do atendimento desses requisitos.

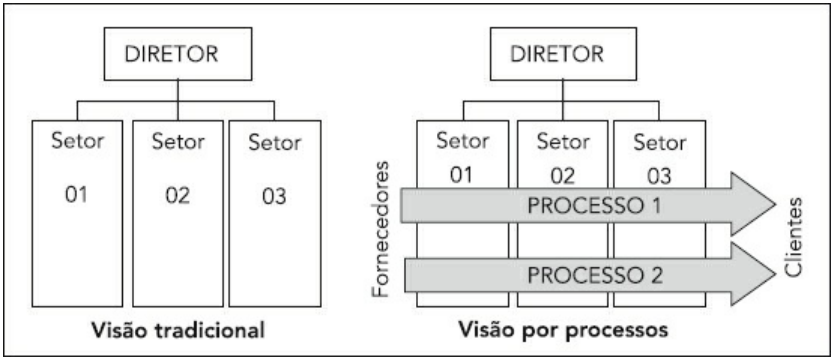

Figura 4 – Visão tradicional e visão por processos.
Fonte: Autor.

Assim, a gestão por processos trata-se do enfoque administrativo aplicado por uma organização que busca a otimização e a melhoria de seus resultados, baseando-se na melhoria e na gestão de seus processos, a partir da mínima utilização de recursos e do máximo índice de acerto.

PARA COMPLEMENTAÇÃO DE ESTUDO

Para que você possa complementar seus conhecimentos sobre processos organizacionais e a gestão por processos, é indicada a leitura do livro *Gestão por Processos*, do autor Saulo Barbara, publicado pela editora Qualitymark em 2006.

CAPÍTULO 2

MAPEAMENTO E MODELAGEM DE PROCESSOS

O mapeamento e modelagem de processos trata-se do conjunto de atividades realizadas para se definir, ou seja, para padronizar e documentar processos organizacionais. Neste capítulo, são apresentados as aplicações e os benefícios do mapeamento e modelagem de processos nas empresas, além das técnicas e ferramentas de apoio utilizadas. Além disso, são apresentadas as principais informações que devem ser documentadas no mapeamento de processos.

2.1 Aplicações e benefícios

A definição de processos, também chamada de mapeamento e modelagem de processos, é realizada por diversas razões, entre as quais podemos citar:

- documentação, análise e otimização de processos;
- padronização na geração de produtos;
- *balanced Scorecard*[1] (BSC);
- gestão de documentos;
- gerenciamento de riscos;
- *benchmarking*[2];
- análise de custos de processos;
- implantação de sistemas ERP[3] (*Enterprise Resourse Planning*);
- certificação em sistema de qualidade (ISO[4], CMMI[5], MPS.BR[6]);
- integração de processos.

Através da definição de processos, as empresas podem obter diversos benefícios, entre os quais destaca-se o registro do capital intelectual da empresa. Assim, através da documentação dos processos, o conhecimento que cada funcionário tem sobre como os processos são executados passa a ficar registrado e sob domínio da organização. Com isso, torna-se mais fácil realizar o treinamento de novos funcionários, já que todas as informações sobre como o trabalho deve ser realizado estão documentadas e disponíveis.

Além disso, a partir da definição dos processos as condições de trabalho são melhoradas, consideravelmente, sendo que todos sabem o que fazer, quando, o que

usar e quais os resultados esperados. De uma forma geral, a empresa fica melhor organizada e os funcionários não têm dúvidas sobre como executar seu trabalho e, se tiverem, sabem onde podem esclarecê-las sem depender de outros (MARANHÃO, 2009).

Durante a definição dos processos, que envolve a participação de todos os envolvidos na discussão dos problemas e possibilidades de melhoria, são identificadas muitas atividades desnecessárias, o que faz com que o custo do processo possa ser reduzido e a produtividade, que se refere à quantidade de produtos produzidos por tempo, seja aumentada.

Além disso, como consequência de todos os benefícios já citados, ocorre um aumento da qualidade dos produtos e da satisfação dos clientes.

2.2 Documentação de processos

As principais informações a serem documentadas sobre um processo, para o seu mapeamento, são: nome, objetivo, políticas e regras, responsável, participantes, ferramenta de apoio, atividades, fluxograma, indicadores de performance e riscos.

Nome – o nome de um processo é a forma pela qual este será referenciado. Exemplos de nomes de processos são: matrícula, venda, análise de crédito, montagem de carro etc.

Objetivo – o objetivo de um processo é a declaração formal e clara de qual é a razão pela qual o processo existe, ou seja, é a declaração de qual é a sua contribuição para o negócio. Por exemplo, o objetivo do processo de matrícula é possibilitar que os alunos da instituição de ensino se inscrevam nas disciplinas que desejam cursar, da forma mais rápida e fácil possível.

Políticas e regras – referem-se às normas que devem ser respeitadas para a execução de um processo. No caso do processo de matrícula, pode-se citar como exemplo a seguinte regra: o aluno só pode se matricular nas disciplinas para as quais já tenha cursado, com aprovação, os pré-requisitos.

Responsável – refere-se ao cargo ou à pessoa responsável pelo processo que está sendo documentado. Por exemplo, podemos definir como responsável pelo processo de matrícula a pessoa que estiver ocupando o cargo de gerente acadêmico da instituição de ensino.

Participantes – os participantes são todos os envolvidos no processo descrito, sendo divididos em três categorias: fornecedores, executores e clientes. Os fornecedores são todos que fornecem as entradas necessárias para a execução do processo, os executores são as pessoas que de fato executam o processo e os clientes são aqueles que recebem as saídas do processo.

Os participantes são, geralmente, representados por papéis que as pessoas

ocupam, cargos ou setores internos das empresas. No caso do processo de matrícula, temos os seguintes participantes: o aluno, que é fornecedor e cliente do processo, visto que fornece informações necessárias para o processo iniciar e recebe uma resposta ao final do processo, além do atendente do setor de matrículas, que é o executor do processo.

Ferramentas de apoio – as ferramentas de apoio são os artefatos utilizados pelos executores para a realização do processo como, por exemplo, um software ou uma calculadora.

Atividades – na documentação de um processo, uma das partes mais importantes é a descrição das atividades que devem ser executadas. Isso porque, além de documentar quais atividades devem ser executadas, devemos documentar também como cada uma delas deve ser realizada, servindo como um guia para os executores do processo. Os nomes das atividades devem iniciar sempre com um verbo no infinitivo, indicando uma ação como, por exemplo, solicitar número de matrícula, verificar pré-requisitos, matricular e imprimir comprovante de matrícula.

Fluxograma – trata-se de um diagrama que ilustra, graficamente, quais são as atividades pertencentes ao processo e a sequência de execução dele, além dos diferentes fluxos possíveis. O fluxograma é uma importante ferramenta para mapeamento, análise e compreensão de processos. A figura 5 apresenta um exemplo de fluxograma.

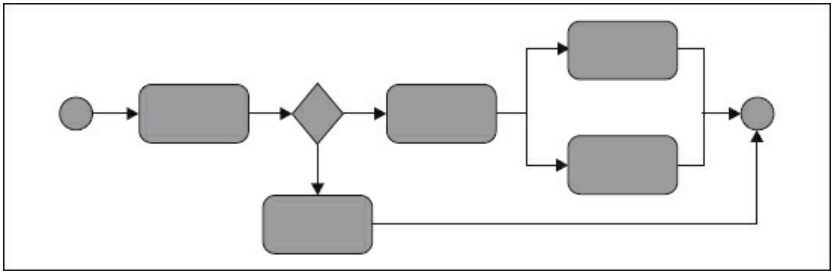

Figura 5 – Exemplo de fluxograma.
Fonte: Autor.

Indicadores de performance – os indicadores de performance, também chamados de KPIs (do inglês *Key Performance Indicators*), são informações, obtidas através de medições nos processos, que permitem monitorá-los durante a sua execução. Esses indicadores são extremamente necessários para possibilitar um adequado gerenciamento dos processos. Contudo, é preciso saber encontrar um equilíbrio entre a falta de indicadores, o que não permite monitorar os processos, e o excesso de indicadores, que pode provocar aumentos dos custos e diminuir a produtividade dos

processos. Para cada indicador de performance devemos registrar as informações apresentadas no quadro 2.

Quadro 2 – Informações sobre os indicadores de performance

Nome	**Nome do indicador**
Responsável	Nome do cargo ou da pessoa responsável pelo indicador.
Fórmula de cálculo	Fórmula para se obter o valor do indicador.
Origem das informações	Informações sobre como obter os dados utilizados no cálculo do indicador.
Meta	Valor meta desejado para o indicador.
Tolerância	Margem de tolerância aceitável para o valor do indicador, em relação à meta.
Periodicidade	Frequência em que o indicador deverá ser obtido.

Um exemplo de um indicador de performance, para o processo de matrícula, é apresentado no quadro 3.

Quadro 3 – Exemplo de indicador de performance

Nome	**Número de disciplinas matriculadas por hora**
Responsável	Gerente acadêmico
Fórmula de cálculo	Somatório do número de disciplinas matriculadas por dia, divido por 10 (horas de atendimento)
Origem das informações	Sistema acadêmico da instituição
Meta	5% do total de disciplinas matriculadas no semestre
Tolerância	Mínimo 3%
Periodicidade	Diária

Riscos – os riscos são os problemas que podem ocorrer durante a execução de um processo. Para cada um dos possíveis riscos devemos registrar as informações apresentadas no quadro 4. Um exemplo de risco para o processo de matrícula é apresentado no quadro 5.

Quadro 4 – Informações sobre os riscos

Nome	**Nome do possível problema**
Responsável	Nome do cargo ou da pessoa responsável pelo possível problema.
Prevenção	Ações a serem tomadas para se evitar que o problema ocorra.
Eliminação	Ações a serem tomadas, caso o problema ocorra de fato.
Indicadores	Indicadores de performance para monitorar o possível problema.
Impacto	Grau de impacto no processo (baixo, médio ou alto), caso o problema ocorra.

Quadro 5 – Exemplo de risco

Nome	**Falha no sistema de matrícula**
Responsável	Gerente de informática
Prevenção	Manutenção e realização de testes periódicos no sistema
Eliminação	Proceder à matrícula através de formulários manuais
Indicadores	➜ Número de disciplinas matriculadas por hora ➜ Tempo de duração da matrícula por aluno
Impacto	Médio

É importante diferenciar os riscos das possiblidades de não conclusão do processo com sucesso, em função das regras de negócio. No processo de matrícula, por exemplo, o fato de o aluno não conseguir se matricular em uma disciplina por falta de um pré-requisito não é um risco e sim um fluxo possível do processo de matrícula. Os riscos são problemas não previstos no fluxo do processo, os quais podem impedir que o processo seja executado.

Com relação ao grau de impacto, geralmente definimos como alto para os problemas que impedem que o objetivo do cliente seja atingido, médio para os problemas que dificultam que o objetivo seja atingido e baixo para os problemas que não são percebidos pelo cliente.

2.3 Técnicas para o mapeamento e modelagem de processos

O mapeamento de processos envolve todas as pessoas de uma organização e provoca muitas mudanças, o que gera, consequentemente, insegurança e resistência em muitos dos colaboradores. Para minimizar esses problemas e aumentar a chance do mapeamento de processo obter sucesso, existem algumas técnicas possíveis de serem aplicadas nas empresas. São elas: entrevistas, observação, questionários, *Business Process Improvement* (BPI), *Joint Application Desing* (JAD), entre outras.

2.3.1 Entrevistas

As entrevistas permitem que a pessoa que está realizando o mapeamento converse com os envolvidos em cada um dos processos da empresa, de tal forma que seja possível compreender como o processo é executado e, a partir disso, documentá-los.

Contudo, a técnica de entrevistas não funciona em todos os casos, pois algumas pessoas apresentam muita dificuldade de explicar, através de uma conversa, como o seu trabalho é realizado no dia a dia. Por isso, existem técnicas alternativas e complementares que permitem uma combinação de diferentes técnicas para o

mapeamento de processos.

2.3.2 Observação

A técnica de observação consiste em realizar uma observação dos funcionários executando os processos de trabalho no dia a dia da empresa. Através dessa técnica, é possível obter informações que nas entrevistas não são explicitadas, por falta de clareza ou por esquecimento do entrevistado.

Na técnica de observação, a pessoa que está mapeando os processos faz anotações sobre o seu funcionamento, enquanto observa o processo ser executado. Além disso, existe a possibilidade de se realizar perguntas aos executantes do processo, durante a observação, a fim de se esclarecer pontos duvidosos. Nesse caso, a técnica de observação é combinada com a de entrevistas.

2.3.3 Questionário

Os questionários são utilizados quando a empresa, na qual se deseja mapear os processos, é grande e distribuída em diversas filiais, distantes uma da outra. Além disso, a técnica de questionário também é útil quando se se deseja obter a opinião de muitas pessoas em um curto espaço de tempo e sem a necessidade de reunir todos os envolvidos.

Para isso, cria-se um documento com um conjunto de perguntas sobre a atual forma de trabalho das pessoas, os problemas que elas identificam e o que elas acham que pode ser melhorado. Posteriormente, distribui-se esse questionário aos funcionários e define-se um prazo para que ele seja respondido. Nos casos em que surgirem dúvidas sobre as respostas, é possível fazer uma entrevista complementar presencialmente, por telefone ou e-mail.

A partir dos questionários, é possível obter-se uma visão geral sobre o que os funcionários pensam sobre os processos da empresa, o que auxilia muito no seu mapeamento.

2.3.4 Business Process Improvement (BPI)

A metodologia BPI, proposta pelo *Business Process Management Institute*[7] (BPMI), tem como objetivo definir as etapas necessárias para um projeto de mapeamento e melhoria de processos, conforme ilustrado na figura 6.

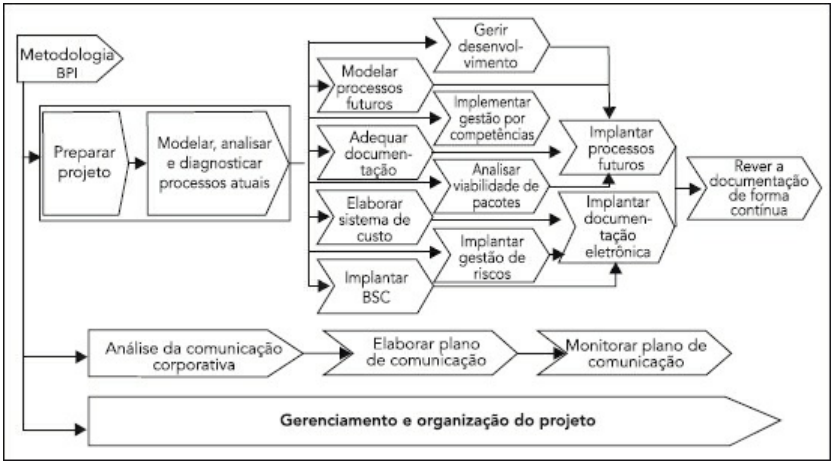

Figura 6 – Metodologia BPI.
Fonte: OMG, 2010.

A metodologia propõe a divisão do projeto em três etapas macro, sendo elas a de planejamento do projeto, mapeamento do processo atual e mapeamento do processo futuro. Além disso, a metodologia propõe atividades de suma importância para o sucesso de um projeto de mapeamento de processos, as quais devem ser executadas durante todo o projeto. São elas a gerência da comunicação e a gerência do próprio projeto.

As etapas da metodologia BPI são listadas abaixo:

1. garantir os recursos necessários para o projeto;
2. levantar os problemas, segundo a ótica dos diretores;
3. estabelecer um software padrão para realizar o mapeamento;
4. conscientizar a alta gestão sobre o projeto;
5. conscientizar operacionais sobre o projeto;
6. coletar e registrar os norteadores;
7. agendar reunião para definição de macroprocessos;
8. modelar macroprocessos atuais;
9. verificar quais processos afetam a satisfação dos clientes;
10. priorizar / definir os processos a serem estudados;
11. definir cronograma do projeto;
12. agendar reuniões com os envolvidos;
13. documentar processos – situação atual;
14. identificar funções críticas;

15. identificar os pontos fracos dos processos atuais;
16. classificar pontos fracos por natureza;
17. obter diagrama de causa e efeito e respectivos indicadores de performance;
18. avaliar impactos de pontos fracos;
19. fundamentar sugestões de melhorias para os pontos fracos;
20. desenhar a proposta de melhoria dos processos;
21. identificar / registrar premissas para mudança;
22. definir os responsáveis dos processos;
23. montar as instruções de serviço das atividades;
24. validar a proposta de mudança dos processos.

2.3.5 Joint Application Design (JAD)

A técnica JAD, criada pela IBM na década de 1970, teve como objetivo inicial reduzir o tempo para a especificação dos sistemas, através da eliminação do retrabalho. Assim, a técnica propôs que os sistemas fossem especificados em grupos de entrevistados formados por todos os envolvidos, direta ou indiretamente, com o processo a ser informatizado. Com isso, os tempos para as especificações diminuíram em mais de 40%.

A parir desses resultados, muitas empresas passaram a adotar o JAD não apenas para especificar sistemas, mas também para todas as atividades que envolvem decisões cujos reflexos podem ter efeitos em vários setores como, por exemplo, o desenvolvimento de novos produtos, reorganização empresarial, normas, processos etc.

A técnica JAD define vários aspectos sobre o mapeamento de processos, como o *layout* da sala onde ocorreram as reuniões em grupo, figura 7, definição das pessoas a serem envolvidas e o roteiro para o encaminhamento das reuniões, entre outros.

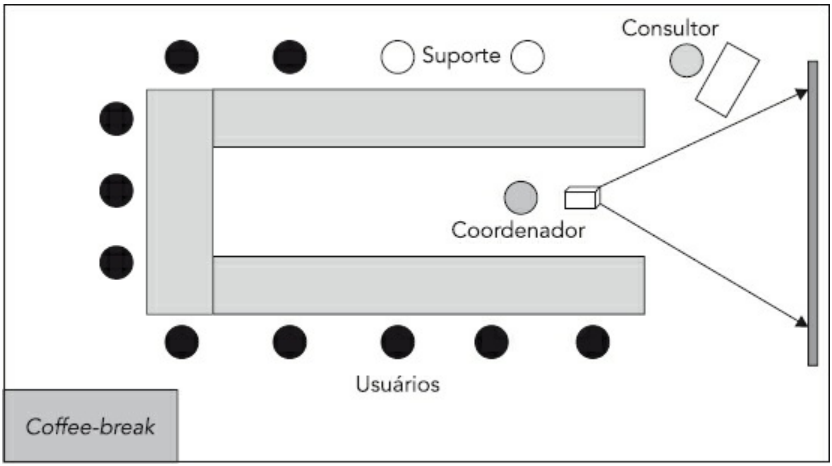

Figura 7 – *Layout* da sala para o mapeamento de processos.
Fonte: Whitten e Bentley, 1999.

Como pode ser observado na figura 7, a técnica JAD define que a mesa de reunião seja disposta em formato de "U", com o objetivo de facilitar a circulação do coordenador e permitir uma maior aproximação com os envolvidos no processo que está sendo discutido.

Além dos envolvidos direta e indiretamente no processo e do coordenador, que tem o papel de mediar as discussões, a técnica define o papel de suporte, geralmente ocupado por pessoas que auxiliam na documentação do processo propriamente dita, além do consultor, que deve ser uma pessoa externa à empresa e com experiência no mapeamento de processos. O consultor atua como uma peça neutra nas discussões, levando o grupo a optar pela melhor solução possível, para cada problema identificado no processo atual.

Outro aspecto muito importante definido pela técnica JAD é o momento do *coffee-break*. A técnica propõe que as reuniões sejam agendadas com antecedência e que tenham uma duração máxima de duas horas, com um intervalo obrigatório. Esse intervalo é extremamente estratégico, pois é o momento menos formal da reunião e, por isso, muitas ideias para solucionar os problemas identificados surgem justamente neste momento.

Além disso, a técnica define que as chefias e gerências não devem participar do mapeamento do processo atual, a fim de evitar que pontos críticos do processo deixem de ser apontados e, também, para evitar que as decisões sofram influência política. Já no mapeamento do processo futuro, a presença dessas pessoas é

extremamente importante, visto que são elas que têm a visão mais ampla da empresa e de seus objetivos estratégicos.

2.4 Ferramentas de apoio

As ferramentas de apoio para o mapeamento de processos dividem-se em duas categorias: ferramentas para criação de fluxogramas e ferramentas para simulação e modelagem de processos.

As ferramentas mais utilizadas para a criação de fluxogramas são apresentadas no quadro abaixo.

Quadro 6 – Principais ferramentas para criação de fluxogramas

Software	Tipo	Fabricante	Site
Argo UML	Gratuita	Argo UML	http://argouml.tigris.org/
BizAgi	Gratuita	BizAgi	http://www.bizagi.com/
Dia	Gratuita	Dia Team	http://projects.gnome.org/dia/
Gliffy	Gratuita	Gliffy Inc	http://www.gliffy.com/
MS Viso	Comercial	Microsoft	http://office.microsoft.com/pt-br/visio/
SmartDraw	Comercial	SmartDraw	http://argouml.tigris.org/

Já as ferramentas para simulação e modelagem de processo são bem mais completas e complexas que as ferramentas para diagramação, visto que permitem simular alterações nos processos, a fim se de verificar o impacto dessas alterações mesmo antes de colocá-las em prática, evitando-se que erros sejam cometidos e a empresa seja prejudicada. O quadro 7 apresenta algumas ferramentas comerciais para simulação de processos.

Quadro 7 – Ferramentas para simulação de processos

Software	Fabricante	Site
ExtendSim	Imagine That Inc.	http://www.extendsim.com/
Arena	Rockwell Automation	http://www.arenasimulation.com/
PowerSim	Powersim Software AS	http://www.powersim.com/
ProModel	ProModel Corporation	http://www.promodel.com/

PARA COMPLEMENTAÇÃO DE ESTUDO

Para aprofundar seu aprendizado sobre mapeamento de processos faça a leitura do livro *Processo Nosso de Cada Dia, Modelagem de Processos de Trabalho*, do autor Mauriti Maranhão publicado pela editora Qualitymark em 2009. Você também pode encontrar muitos artigos sobre esse assunto no portal da administração (http://www.administradores.com.br).

CAPÍTULO 3

NOTAÇÕES PARA MAPEAMENTO DE PROCESSOS

Este capítulo apresenta as duas principais notações utilizadas para a criação de fluxogramas de processos: UML e BPMN. A *Unified Modeling Language*[8] (UML) trata-se de uma linguagem padrão para modelagem de software que possui diversos diagramas, dentre eles o diagrama de atividades, cujo objetivo é a modelagem de processos. Além da UML, é apresentada a *Business Process Modeling Notation*[9] (BPMN), uma notação amplamente utilizada para o mapeamento de processos de negócios.

Tanto a UML como a BPMN têm como objetivo permitir a fácil compreensão de processos de trabalho, através da sua representação em diagramas, por executivos de negócio, desenvolvedores de software e todas as demais pessoas que irão interagir, monitorar e gerir os processos.

3.1 *Unified Modeling Language* – UML

A Linguagem Unificada de Modelagem (UML) originou-se a partir da padronização das metodologias de desenvolvimento de sistemas baseados na orientação a objetos[10]. Foi criada por três grandes desenvolvedores de sistemas orientados a objetos: Grady Booch, James Rumbaugh e Ivar Jacobson (FOWLER, 2005).

3.1.1 Objetivos e aplicações

A UML é uma linguagem gráfica para visualização, especificação, construção e documentação de software. Ela se propõe a ser a linguagem padrão para modelagem de sistemas orientados a objetos, por ser unificada e facilitar que os profissionais da área interpretem, de uma maneira correta e sem ambiguidades, os modelos gerados (OMG, 2003).

A UML é uma linguagem visual, pois trata-se de uma notação diagramática padrão para desenhar ou apresentar figuras, com algum texto, relacionadas ao desenvolvimento de software (LARMAN, 2007).

No contexto deste livro, abordaremos apenas o diagrama de atividades da UML utilizado para a modelagem de fluxos de processos.

3.1.2 Diagrama de atividades e seus elementos

Um diagrama de atividades UML mostra atividades sequenciais e paralelas em um processo. Eles são úteis para a modelagem de processos de negócios, fluxos de trabalho, fluxo de dados e algoritmos complexos (LARMAN, 2007).

O diagrama de atividades oferece uma notação rica para mostrar uma sequência de atividades, inclusive paralelas, e pode ser aplicado em qualquer perspectiva ou propósito, sendo mais utilizado para visualizar fluxos de trabalho e processos de negócios.

Conforme ilustrado na figura 8, os principais elementos de um diagrama de atividades são: início do processo, atividades, decisões, transições, barras de sincronização, partições e o fim do processo.

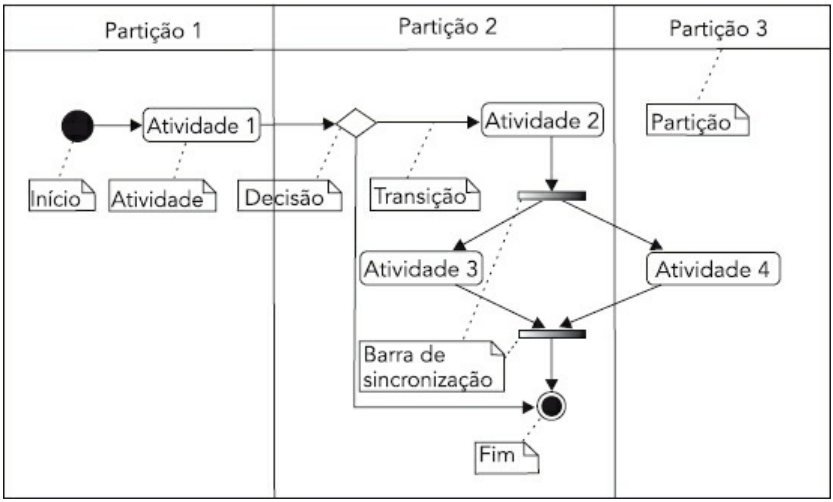

Figura 8 – Diagrama de atividades da UML.
Fonte: Autor.

Estado inicial – indica o início do processo.

Atividade – representa uma ação executada no processo.

Transição – indica o fluxo do processo.

Decisão – indica diferentes possibilidades de transição. A partir de uma decisão, o processo segue por um ou outro fluxo.

Barra de sincronização – indica paralelismo na execução das atividades. No

exemplo da figura 8, após a execução da atividade 2, as atividades 3 e 4 são executadas paralelamente e o fim do processo só é atingido quando as duas forem concluídas.

Partição – também chamada de raia, representa os setores ou papéis envolvidos no processo.

Estado final – indica o final de um processo.

3.1.3 Exemplo

A figura 9 apresenta um exemplo de um processo representado através de um diagrama de classes UML, criado através da ferramenta gratuita Astah Community[11]. Trata-se do processo de troca de turma, através do qual o atendente solicita informações do aluno que deseja realizar a troca, registra o pedido e verifica se há vaga na turma pretendida. Caso não existam vagas disponíveis, o atendente encerra o pedido e envia uma notificação para o e-mail do aluno.

Caso exista vaga e a solicitação tenha sido feita dentro do período autorizado, o atendente realiza a troca, encerra o pedido e envia a notificação ao aluno. Caso exista a vaga, mas a solicitação esteja fora do período autorizado, o atendente encaminha a solicitação para uma analise do coordenador do curso. Com a autorização da coordenação, o atendente realiza a troca. Caso contrário, o coordenador registra a justificativa para a não autorização e o atendente encerra o pedido e notifica o aluno.

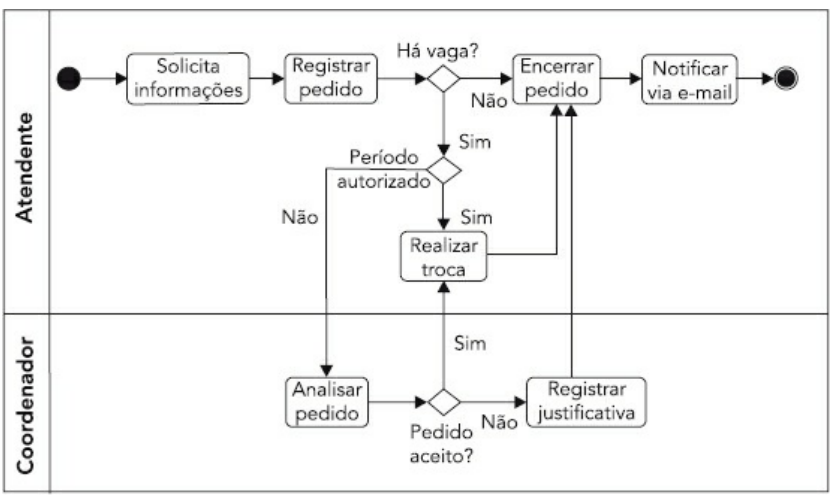

Figura 9 – Diagrama de atividades do processo de troca de turma.
Fonte: Autor.

3.2 *Business Process Management Notation* – BPMN

A notação BPMN foi desenvolvida pelo *Business Process Management Initiative*[12] (BPMI) com o objetivo de ser um padrão para modelagem de processos de negócios. Sua primeira versão foi disponibilizada em maio de 2004, após dois anos de estudos do grupo BPMI *Notation Working Group*, com o objetivo de prover uma notação que fosse rapidamente entendida por todos os usuários de negócio.

3.2.1 Objetivo e aplicações

A BPMN define um diagrama de processos de negócio baseado em técnicas de fluxograma. Além disso, foi desenvolvida, também, para permitir a geração de executáveis através de seu modelo.

A notação apoia apenas os conceitos de modelagem que são aplicáveis aos processos de negócio. Não faz parte da BPMN:

- estrutura organizacional e recursos;
- estrutura funcional de organizações;
- dados e informações;
- estratégia;
- regras de negócio.

O objetivo principal desta notação é a especificação e modelagem de processos e permitir a simulação de um processo modelado para sua verificação.

3.2.2 Elementos

A BPMN oferece uma notação com diversos elementos para o mapeamento de processos de negócio. A figura 10 apresenta os elementos principais da notação, muitos deles iguais ou semelhantes aos elementos do diagrama de atividades da UML.

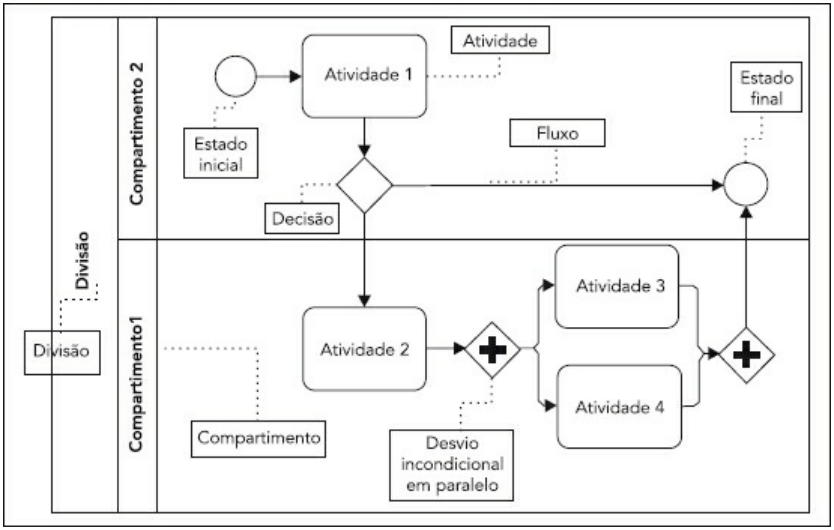

Figura 10 – Elementos principais da BPMN.
Fonte: Autor.

Os elementos básicos da BPMN são: início, fim, atividades, decisões, fluxo, desvio incondicional em paralelo, divisões e compartimentos.

Estado inicial – indica o início do processo.

Fluxo – indica o fluxo de execução do processo, ou seja, a transição entre as atividades do processo.

Atividade – representam as ações que devem ser executadas pelo processo. **Decisão** – indica uma condição para a definição do fluxo do processo a ser seguido. Apenas um dos fluxos é escolhido cada vez que o processo passa por uma decisão.

Desvio incondicional em paralelo – indica que a partir deste ponto o processo segue para a execução de atividades em paralelo. Somente após a finalização de todas as atividades executadas em paralelo o processo segue a continuação do seu fluxo.

Divisão – representa os setores ou papéis responsáveis pela execução das atividades do processo.

Compartimento – representa um processo.

Estado final – indica a finalização do processo.

3.2.3 Exemplo

A figura 11 apresenta um exemplo de um processo representado através de um diagrama construído com a notação BPMN. O diagrama foi criado através da ferramenta gratuita BizAgi[13] e trata-se do mesmo processo de troca de turma apresentado no diagrama de atividades da UML.

Nesse processo, o atendente solicita informações do aluno que deseja realizar a troca, registra o pedido e verifica se há vaga na turma pretendida. Caso não existam vagas disponíveis, o atendente encerra o pedido e envia uma notificação para o e-mail do aluno. Caso exista vaga e a solicitação tenha sido feita dentro do período autorizado, o atendente realiza a troca, encerra o pedido e envia a notificação ao aluno. Caso exista a vaga, mas a solicitação esteja fora do período autorizado, o atendente encaminha a solicitação para uma análise do coordenador do curso. Com a autorização da coordenação o atendente realiza a troca. Caso contrário, o coordenador registra a justificativa para a não autorização e o atendente encerra o pedido e notifica o aluno.

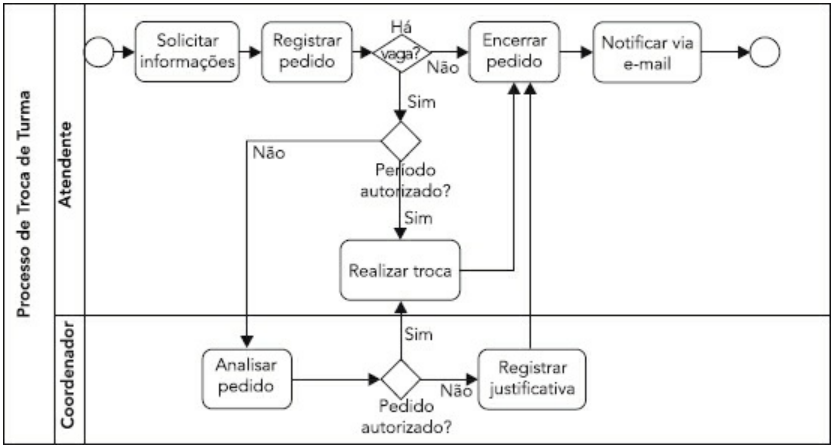

Figura 11 – Diagrama BPMN do processo de troca de turma.
Fonte: Autor.

PARA COMPLEMENTAÇÃO DE ESTUDO

Para saber mais sobre a notação UML, acesse o site www.uml.org ou consulte o livro

UML Essencial, do autor Martin Fowler, publicado pela editora Bookman em 2005. Para mais informações sobre a BPMN, consulte o site www.bpmn.org ou o livro *BPMN 2.0*, do autor Thomas Allweyer.

CAPÍTULO 4

AVALIAÇÃO E MUDANÇA DE PROCESSOS

Este capítulo apresenta os princípios básicos para a avaliação e mudança de processos, etapas fundamentais em um projeto de definição de processos. Assim, são apresentadas as fases para a avaliação e mudança de um processo, bem como as premissas essenciais para que o novo processo seja realmente executado por todos os colaboradores da empresa.

O processo de avaliação e mudança tem como objetivo melhorar as empresas através da identificação dos problemas críticos e determinação das prioridades de melhoria, sendo seus objetivos compreender o processo atual, identificar os problemas principais, realizar as melhorias necessárias e controlar para que todos sigam os novos processos.

4.1 Princípios básicos

Alguns princípios básicos para que a avaliação e mudança de um processo obtenha sucesso são:

- **definir o modelo de processo desejado**, ou seja, definir previamente as estratégias para a mudança, de tal forma que já se tenha algum norteador durante a avaliação do processo atual e, consequentemente, maiores chances de atingir os resultados desejados;
- **garantir sigilo**, de forma a estimular as pessoas a falarem abertamente sobre os problemas, sem receio de prejudicar colegas de trabalho. É muito importante esclarecer, a todos os envolvidos, que o objetivo do projeto é identificar problemas nos processos e não nas pessoas;
- **envolver a alta gerência**, a fim de garantir que o projeto terá prioridade, ou seja, que as pessoas se organizarão para participar das reuniões e que estarão comprometidas em ajudar. Além disso, o envolvimento da alta administração é fundamental para que as melhorias sejam, de fato, colocadas em prática;
- **garantir respeito dos funcionários**, ou seja, as pessoas escolhidas para participar do grupo de melhoria de processos devem ser experientes e respeitadas pelos colegas, a fim de atuarem como influenciadores e motivadores do projeto, além de formadores de opinião;
- **focar na ação**. As melhorias propostas devem ser realistas a tal ponto que sejam cabíveis de execução, visto que o principal objetivo do projeto é a

melhoria dos processos;

- **desenvolver boa liderança**, a fim de que as discussões sejam rapidamente encaminhadas para a melhor solução e que as pessoas sintam-se motivadas e confiantes com o projeto;
- **utilizar consultoria externa**, a qual tem como objetivo atuar como conselheira e apaziguadora nas discussões, influenciando as pessoas a tomarem a melhor decisão para os objetivos estratégicos da organização, baseado nas melhores práticas do mercado.

4.2 Fases

O processo de avaliação e mudança divide-se em cinco fases: preparação, avaliação, recomendações, mudança e manutenção.

4.2.1 Preparação

A fase de preparação, com duração de um ou dois dias, envolve todas as ações necessárias para preparar a empresa para o projeto de melhoria de processos. Assim, inicialmente é necessário obter o comprometimento da alta administração com o projeto, sem o qual dificilmente um projeto de mudança de processos terá sucesso. A alta administração deve se comprometer em priorizar, participar e executar as mudanças acordadas durante o projeto de melhoria de processos.

Nesta fase, deve-se, também, definir e treinar a equipe que conduzirá o trabalho, a qual deve ser composta por duas ou três pessoas, com dedicação exclusiva ao projeto. Essa equipe deve ser composta por funcionários experientes, com conhecimentos profundos sobre o processo atual da empresa, bem como formadores de opinião, de tal forma que influenciem positivamente os demais colaboradores da organização.

Além disso, a equipe deve contar com um consultor externo, com conhecimento e experiência em mudança de processos, a fim de atuar como um conselheiro e influenciador do processo decisório. A participação de um consultor externo é essencial, pois ele não tem interesses políticos e não pertence à hierarquia da empresa e, assim, poderá conduzir as decisões sem qualquer receio.

4.2.2 Avaliação

Trata-se da fase em que o processo atual é avaliado, através do seu mapeamento,

que é realizado em grupos de discussão envolvendo representantes de todas as pessoas envolvidas direta ou indiretamente no processo.

Nesta fase, cada um dos processos é compreendido, documentado, e os seus problemas são identificados. Assim, é muito importante que as chefias e gerências não participem diretamente, pois desta forma os participantes ficam mais à vontade para apontar os problemas e sugerir melhorias.

Essa fase tem duração prolongada e é concluída com a entrega dos problemas identificados para a alta administração da empresa. A avaliação inicia com a apresentação do líder e da equipe do projeto, bem como dos objetivos, princípios e cronograma, para todos os colaboradores da empresa. É muito importante que todos conheçam e compreendam bem o projeto e que a alta administração solicite o comprometimento e a colaboração de todos.

Após, devem ser agendadas reuniões específicas com os gerentes dos processos a serem avaliados e com os funcionários a serem entrevistados. Deve ser esclarecido que o foco da avaliação é compreender o que está sendo feito atualmente, como está sendo feito e quais são os problemas enfrentados.

Antes de iniciar a avaliação de fato, deve-se preparar um conjunto de questões prévias sobre o processo, as quais servem como base para as reuniões com o grupo de pessoas envolvidas no processo. Além das reuniões, é importante escutar as pessoas em conversas informais, nas quais, muitas vezes, obtêm-se melhores informações sobre os problemas e sugestões para resolvê-los.

As reuniões devem envolver entre seis e oito pessoas, executoras do processo e favoráveis ao projeto de mudança, com duração entre 90 e 120 minutos. No final da avaliação, deve-se apresentar os problemas identificados, as consequências dos mesmos e exemplos, sem citar pessoas ou projetos específicos. O relatório final deve conter uma descrição resumida do processo de avaliação, bem como os problemas encontrados e o plano de ação para cada um deles.

4.2.3 Recomendação

Nesta fase, elabora-se o plano de ação para cada um dos problemas identificados, com base nas sugestões de melhoria apontadas na avaliação do processo. O plano de ação define quem, quando e como as melhorias serão realizadas. Nesse momento, a participação das chefias e gerências é indispensável, visto que essas pessoas possuem conhecimento estratégico da empresa, ou seja, sabem qual o direcionamento que a empresa deseja para o futuro, a médio e longo prazos.

4.2.4 Mudança

Essa é a etapa em que o plano de ação, para cada problema identificado na avaliação, é colocado em prática. Para isso, deve-se priorizar os problemas e, consequentemente, dividir as ações em curto, médio e longo prazos. Os problemas de maior prioridade são aqueles que afetam diretamente a satisfação do cliente.

O comprometimento da alta administração com a execução das mudanças é um dos fatores fundamentais para um projeto de melhoria de processo. Caso as mudanças não sejam executadas, muitos funcionários ficarão frustrados e desmotivados com o projeto e dificilmente se comprometerão em um novo projeto de melhoria.

4.2.5 Manutenção

A manutenção tem como objetivo controlar para que o novo processo seja realmente executado por todos. É muito comum as pessoas terem resistência a mudanças e, caso não sejam controladas, facilmente voltam a realizar o seu trabalho da forma como estavam acostumadas. Além disso, é na fase de manutenção que devemos verificar se o novo processo é realmente adequado ou se necessita de ajustes, além da identificação de novas oportunidades de melhoria.

Para que o novo processo seja de fato executado e evite-se que aos poucos os processos antigos voltem a ser realizados, é necessário manter a equipe de melhoria, visto que se trata do ponto de referência para o esclarecimento de dúvidas e é grande incentivadora da adoção do novo processo. É necessário, também, controlar continuamente as pessoas, através de auditorias, para certificar-se de que elas estão utilizando os novos processos e estabelecer um programa de treinamento, a fim de capacitar os novos funcionários nos processos da empresa.

PARA COMPLEMENTAÇÃO DE ESTUDO

Para obter mais informações sobre avaliação e mudança de processos consulte a norma NBR ISO/IEC 15504:2008, que trata especificamente da avaliação de processos de Tecnologia da Informação.

CAPÍTULO 5

PROCESSO DE SOFTWARE

Este capítulo apresenta o conceito de processo de software, bem como suas metas e as condições para que sejam alcançadas com sucesso. Além disso, são apresentadas suas atividades fundamentais e de apoio.

5.1 Conceito, objetivos e condições

O processo de software é definido como um conjunto de atividades, métodos, práticas e transformações que pessoas empregam para desenvolver e manter software e produtos associados (HUMPHREY, 2004).

Segundo Pressman (2001), o processo de software envolve a definição dos métodos, ferramentas e atividades necessárias para o desenvolvimento e a manutenção de software, sendo os métodos definidos como a forma com que o processo é executado, as ferramentas sendo aquelas que oferecem apoio automatizado ou semiautomatizado aos métodos, aos processos e às atividades necessárias para que o software seja construído e mantido.

Já Sommerville (2005) define processo de software como um conjunto de atividades e resultados associados que levam à produção de um produto de software. Esse processo pode envolver o desenvolvimento de software desde o início, embora, cada vez mais, ocorra o caso de um software novo ser desenvolvido mediante a expansão e a modificação de sistemas já existentes.

Quanto aos objetivos, pode-se definir que todo o projeto de software deve entregar um sistema que atenda as necessidades e expectativas dos usuários, cumprindo as seguintes metas, definidas previamente:

- prazo;
- custo;
- qualidade.

Contudo, sabe-se que os projetos de software são extremamente complexos e envolvem muitas variáveis que, se não devidamente controladas, acarretam o insucesso do projeto.

Assim, após inúmeros projetos malsucedidos, com resultados inadequados, tanto

em relação a custo e prazo como em relação à qualidade final, muitas empresas perceberam a importância de definir seus processos e utilizar métodos e ferramentas adequadas.

Por isso, conforme ilustrado na figura 12, as três condições fundamentais para que um projeto de software obtenha sucesso são: estabelecer uma equipe com pessoas capacitadas e motivadas, possuir processos de trabalho definidos e seguidos por todos e disponibilizar ferramentas de apoio adequadas.

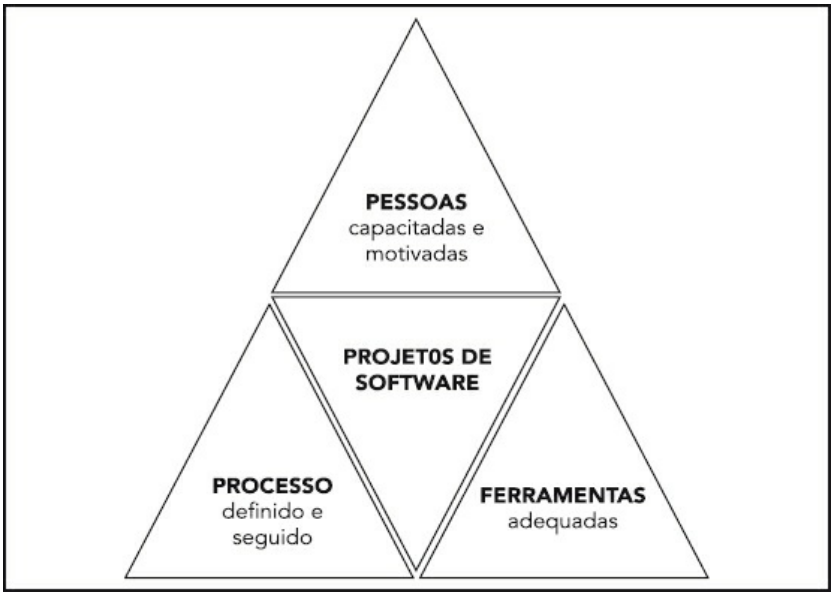

Figura 12 – Condições para projetos de software bem-sucedidos.
Fonte: Autor.

Além de possuir uma equipe com a quantidade de pessoas necessária, é preciso que todos os membros da equipe estejam motivados e capacitados para realizar o trabalho, tanto do aspecto técnico como processual. Para isso, deve-se garantir que todos já possuam o conhecimento técnico e sobre o processo de trabalho ou que sejam treinados previamente.

Com relação à motivação, deve-se oferecer condições de trabalho e benefícios suficientes para que as pessoas estejam satisfeitas com seu atual emprego, minimizando as chances de resultados de baixa qualidade e a perda de membros da equipe durante a execução do projeto.

Todavia, apenas pessoas capacitadas e motivadas não são suficientes para que um projeto de software obtenha sucesso. É preciso que existam processos de trabalho definidos, os quais facilitam e organizam o trabalho da equipe, e que esses processos sejam de fato seguidos por todos. Como já mencionado no Capítulo 4, para que um processo seja seguido se faz necessária a realização de auditorias periódicas, as quais garantem que o processo definido esteja realmente sendo executado por todos.

Para que o projeto tenha produtividade e se atinja a qualidade esperada, são necessárias ferramentas de apoio adequadas. Atualmente, existem muitas ferramentas de apoio ao processo de software, disponíveis tanto comercialmente como gratuitas. Assim, o desafio para as empresas é definir quais são as ferramentas melhor adequadas para a suas características e necessidades.

Contudo, não há um processo de software ideal e, assim, diferentes empresas definem abordagens inteiramente diferentes para o desenvolvimento de software. Em todas as empresas os processos evoluem para explorar a capacidade das pessoas, assim como em função do tipo de sistema que é desenvolvido.

Por isso, diferentes autores classificam as atividades do processo de software de diversas formas. Para Sommerville (2005), embora existam muitos processos de software diferentes, há atividades fundamentais comuns a todos eles. São elas: *Especificação*, na qual são definidas as suas funcionalidades e restrições; *Projeto e Implementação*, na qual o software é produzido de acordo com a especificação; *Validação*, na qual o software é validado para garantir que faz o que o cliente deseja, e a *Evolução*, na qual o software é adaptado para atender às necessidades mutáveis do cliente.

Nesse livro, as atividades do processo de software foram classificadas em fundamentais e de apoio. As atividades fundamentais representam as atividades técnicas necessárias para a construção e manutenção de software, sendo elas análise, projeto, implementação, testes, implantação e manutenção. Já as atividades de apoio são atividades necessárias para garantir que as atividades fundamentais sejam realizadas da melhor forma possível, sendo elas a gerência de projetos, a gerência da configuração e a gerência da qualidade.

5.2 Atividades fundamentais

Como já mencionado anteriormente, as atividades fundamentais do processo de software são as atividades técnicas necessárias para que o software seja construído e mantido.

5.2.1 Análise

A análise, também chamada de análise de sistemas ou especificação de sistemas, trata-se da primeira e fundamental atividade do processo de software. É através desta atividade que o analista de sistemas, em conjunto com o cliente e os usuários, define as funcionalidades e restrições do sistema que será desenvolvido, o que chamamos de requisitos do sistema.

Para isso, o analista deve obter o maior número possível de informações, como os objetivos de negócio do cliente, as necessidades dos usuários, os sistemas já existentes na empresa, as regras e os regulamentos da área, além de informações sobre a área de negócio para a qual o sistema será desenvolvido. A figura 13 ilustra o processo de análise de um sistema, apresentando suas entradas e saídas.

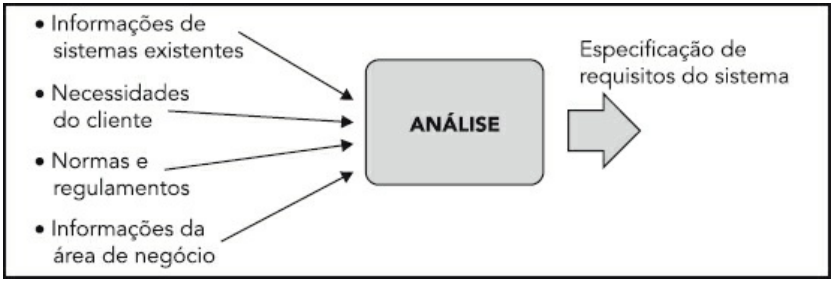

Figura 13 – Processo de análise de um sistema.
Fonte: Adaptado de Rotonya e Sommerville,1998.

Quanto maior o número de requisitos identificados e especificados corretamente nessa etapa, melhores são as estimativas de prazo e custo e maiores são as chances do software ser desenvolvido com qualidade e atender às necessidades e expectativas do cliente.

Entretanto, isso não impede que muitos requisitos sejam identificados durante a construção do sistema, o que de certa forma facilita a identificação de novas funcionalidades e restrições, visto que tanto o cliente como a equipe de desenvolvimento obtêm uma melhor compreensão do sistema que está sendo desenvolvido.

Como resultado da atividade de análise, temos um documento com a descrição detalhada, em linguagem de usuário, de cada uma das funcionalidades e restrições do sistema. Esse documento é comumente chamado de Especificação de Requisitos do Sistema.

A análise de um sistema subdivide-se em cinco atividades específicas. A figura 14 ilustra as cinco atividades, sendo elas elicitação, análise/negociação, documentação, validação e gerenciamento de requisitos.

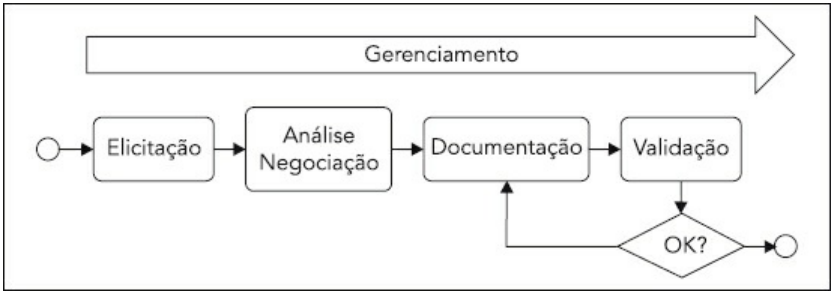

Figura 14 – Atividades da etapa de análise de um sistema.
Fonte: Autor.

Elicitação de requisitos é o nome dado para a atividade de identificação, de descoberta dos requisitos necessários para o desenvolvimento de um sistema. Analistas trabalham junto com clientes e usuários para entender o problema e as necessidades e identificar o maior número possível de requisitos para o projeto.

Posteriormente, ou até mesmo em paralelo com a elicitação, o analista de sistemas realiza a análise dos requisitos identificados com o objetivo de verificar se existem requisitos faltantes, requisitos desnecessários ou requisitos redundantes. Nesse caso, os problemas identificados pelo analista são negociados com o cliente, a fim de se excluir os requisitos desnecessários e redundantes e incluir os requisitos faltantes, fechando-se, assim, o escopo inicial do projeto.

Somente após o fechamento do escopo, ou seja, da definição final da lista de requisitos do projeto, é que se inicia a documentação de tais requisitos. Nessa etapa, cada requisito é detalhadamente descrito de tal forma que o cliente possa validá-los e que a equipe de desenvolvimento possa compreendê-los, a fim de construí-los da forma mais correta possível.

Assim que a documentação dos requisitos estiver concluída, inicia-se a etapa de validação, através da qual uma ou mais pessoas verificam o documento com o objetivo de certificar-se de que cada requisito está claramente e corretamente descrito. Essa é uma etapa fundamental, visto que é através dela que o cliente verifica se a solução que o analista de sistemas elaborou, para a sua necessidade, está ou não de acordo. É através da validação dos requisitos que garantimos que o cliente saberá exatamente que sistema receberá ao final do projeto.

Em paralelo com as atividades anteriores, e durante todo o desenvolvimento e uso do sistema, devemos realizar o gerenciamento dos requisitos. Essa atividade é fundamental para que toda e qualquer mudança necessária no sistema seja devidamente documentada e controlada, evitando-se que ao alterarmos um requisito outros sejam indevidamente afetados.

5.2.2 Projeto

A etapa de projeto de um sistema, também chamada de engenharia ou arquitetura de software, é a etapa na qual o projetista, engenheiro ou arquiteto do sistema projeta uma estrutura de software que atenda da melhor forma possível à especificação de requisitos.

Um projeto de software é uma descrição da estrutura de software a ser construída, da estrutura dos dados a serem armazenados e processados, das interfaces entre os componentes do sistema e, algumas vezes, dos algoritmos a serem utilizados.

A etapa de projeto pode envolver o desenvolvimento de vários modelos do sistema, em diferentes níveis de abstração, podendo ser uma especificação abstrata, produzida para esclarecer os requisitos, ou uma especificação mais detalhada e precisa de como o sistema deve ser estruturado. Assim, alguns exemplos de itens projetados nesta etapa são:

- *arquitetura do sistema* – os subsistemas que constituem o sistema e suas relações são definidos e documentados;
- *subsistemas* – para cada subsistema é projetada uma estrutura que contemple suas funções e restrições;
- *interfaces* – para cada subsistema, sua interface com outros subsistemas é projetada e documentada;
- *componentes* – são projetados os componentes do sistema que contemplarão as diversas funções especificadas nos requisitos; cada componente possuirá um conjunto de funções do sistema;
- *estrutura de dados* – a estrutura para o armazenamento dos dados do sistema é detalhadamente projetada e documentada;
- *algoritmos* – os algoritmos para os serviços mais complexos são projetados.

Apesar disso, é muito comum, nas organizações, que o desenvolvimento dos sistemas inicie a partir de um conjunto prévio de requisitos, para o qual um projeto informal é preparado. Após, esse projeto é modificado e complementado à medida que o sistema é desenvolvido e novos requisitos são definidos. Neste caso, se não houver um adequado controle dessas mudanças, ao final, o projeto documentado é uma descrição incoerente e incompleta do sistema.

Uma abordagem mais metódica para a etapa de projeto de sistemas é proposta pelos métodos estruturados, que são conjuntos de notações e diretrizes para o projeto de software. Alguns exemplos de métodos estruturados para o projeto de sistemas são: Projeto Estruturado (CONSTANTINE e YORDOUN, 1979), Análise de Sistemas Estruturados (GANE e SARSON, 1979) e Desenvolvimento de Sistemas (JACKSON, 1983). Além desses, um dos métodos mais utilizados, atualmente, é o do projeto orientado a objetos (ROBINSON, 1992; BOOCH, 1994; RUMBAUGH *et al.*, 1991; BOOCH *et al.*,1999).

Embora esse livro não tenha como objetivo apresentar os métodos de projeto de software, a seguir são ilustrados exemplos dos possíveis diagramas elaborados em projetos orientados a objetos, através da notação UML.

A figura 15 apresenta o diagrama de casos de uso, que tem como objetivo auxiliar a comunicação entre os analistas e o cliente. Esse diagrama apresenta as funcionalidades do sistema, do ponto de vista do cliente.

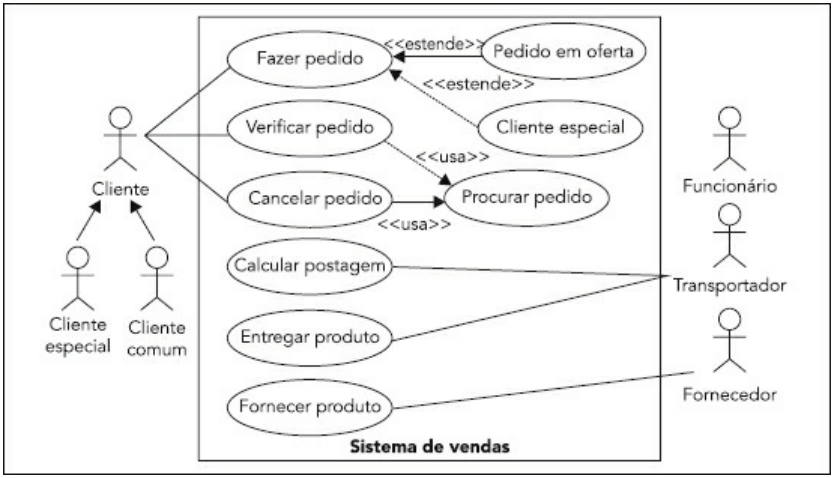

Figura 15 – Exemplo de um diagrama de casos de uso.
Fonte: Autor.

Um dos diagramas mais utilizados para projetar software orientado a objetos é o diagrama de classes, ilustrado na figura 16, que apresenta os diversos tipos de objetos do sistema e o relacionamento entre eles.

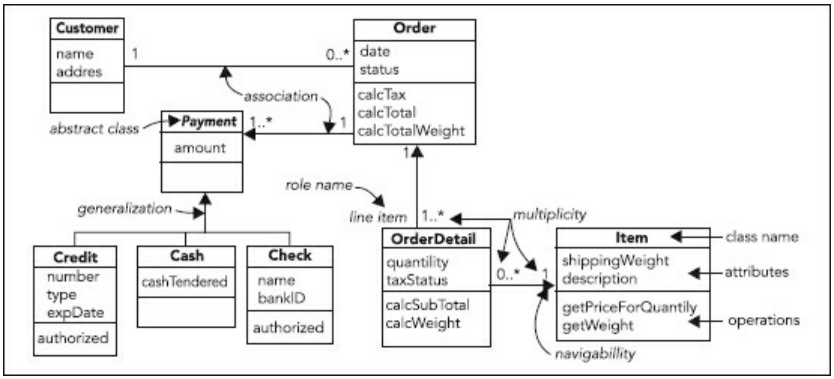

Figura 16 – Diagrama de classes.
Fonte: www.distancelearnig101.net/interm_prog/uml/.

A figura 17 apresenta o diagrama de sequência, o qual tem como objetivo mostrar como as mensagens entre os objetos são trocadas, no decorrer do tempo, para a realização de uma operação.

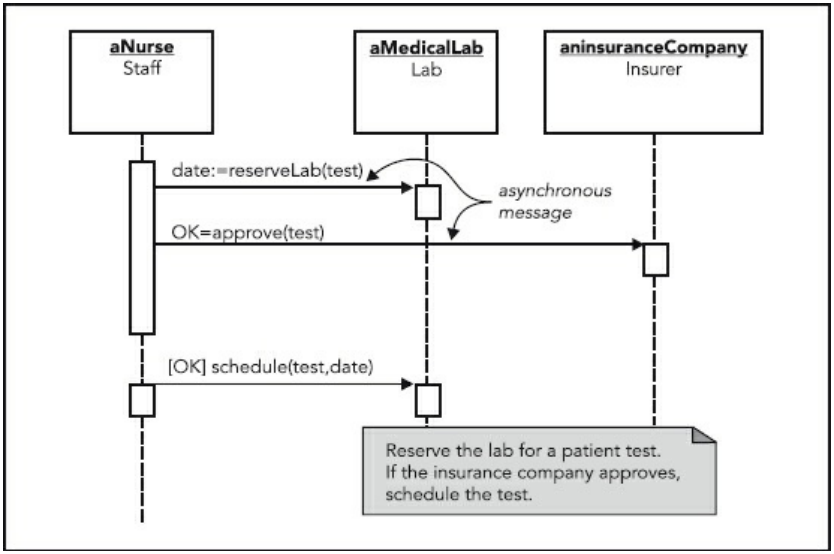

Figura 17 – Exemplo de diagrama de sequência.
Fonte: www.distancelearnig101.net/interm_prog/uml/.

Já a figura 18 ilustra um diagrama de colaboração, o qual tem o mesmo objetivo do diagrama de sequência, porém neste o tempo não é mais representado por linhas verticais, mas sim através de uma numeração.

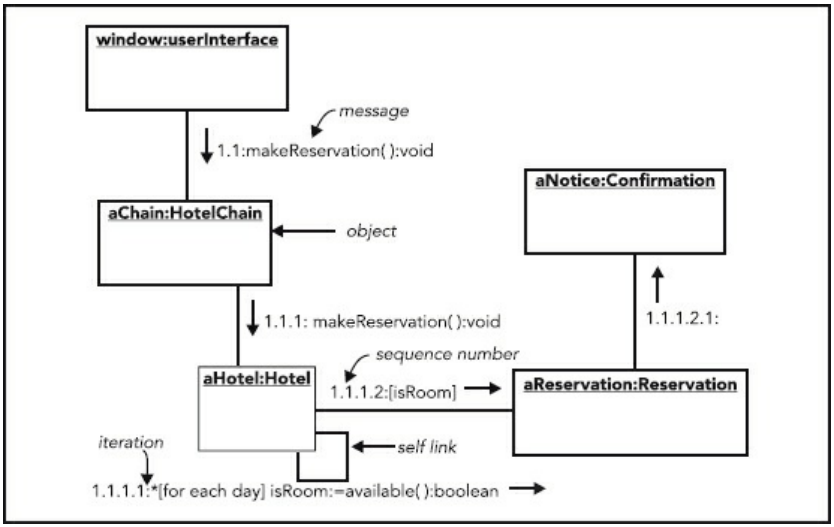

Figura 18 – Diagrama de colaboração.
Fonte: www.distancelearnig101.net/interm_prog/uml/.

O diagrama de estado, apresentado na figura 19, mostra os possíveis estados de um objeto e as transações responsáveis pelas suas mudanças de estado.

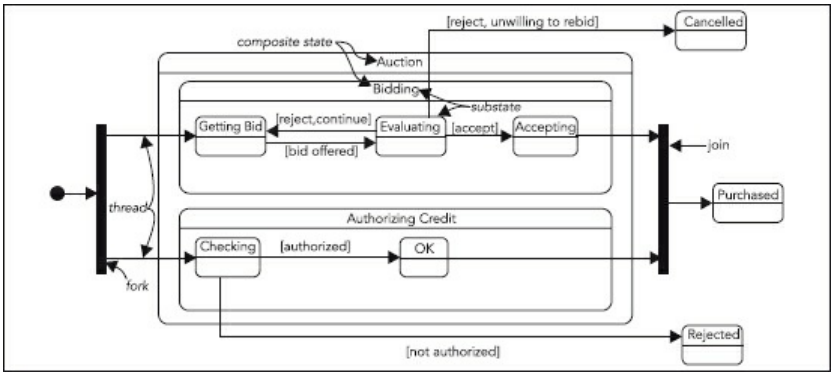

Figura 19 – Diagrama de estado.
Fonte: www.distancelearnig101.net/interm_prog/uml/.

Como já vimos no Capítulo 3, o diagrama de atividade, apresentado na figura 20, tem como objetivo mostrar o fluxo de atividades em um processo.

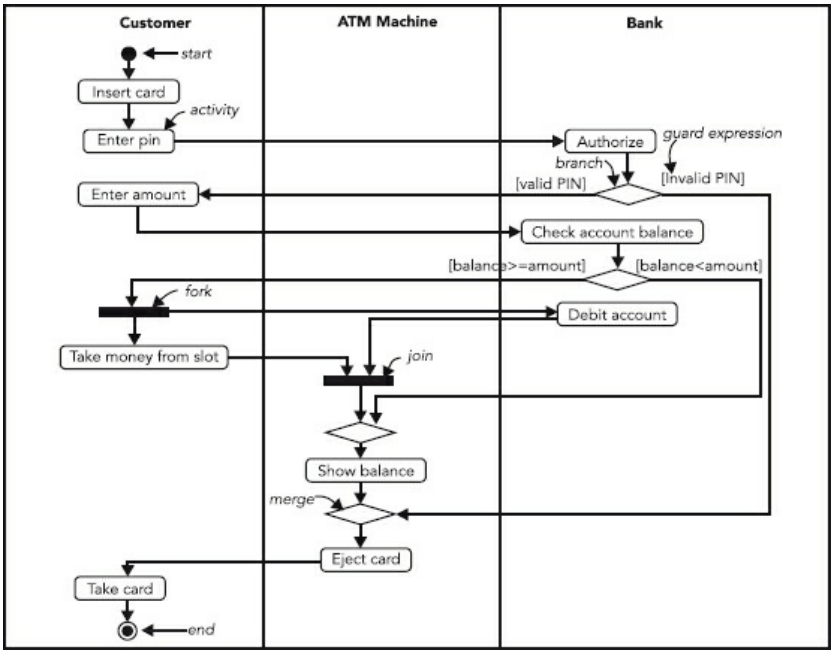

Figura 20 – Diagrama de atividades.
Fonte: www.distancelearnig101.net/interm_prog/uml/.

A figura 21 apresenta um exemplo do diagrama de componentes, o qual tem como objetivo definir o agrupamento das classes de um sistema em pacotes.

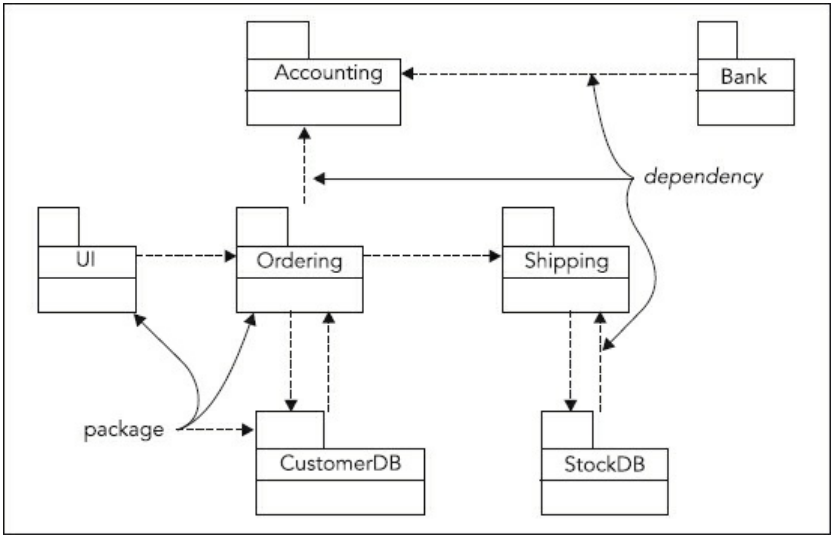

Figura 21 – Diagrama de componentes.
Fonte: www.distancelearnig101.net/interm_prog/uml/.

Finalmente, a figura 22 apresenta o diagrama de implantação que mostra a configuração de nós de processamento, em tempo de execução, e os componentes que neles existem.

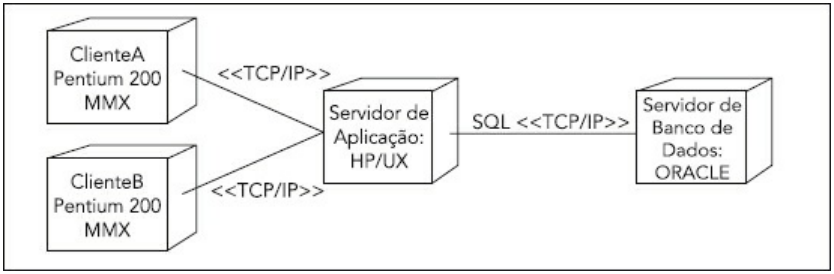

Figura 22 – Exemplo de diagrama de implantação.
Fonte: www.distancelearnig101.net/interm_prog/uml/.

5.2.3 Implementação

A implementação trata-se da etapa de conversão de um conjunto de especificações em sistema executável. É a etapa em que o sistema é de fato construído através da sua programação. A implementação é um processo, geralmente individual, que deve seguir determinados padrões para que produza resultados de fácil manutenibilidade.

Para a realização dessa atividade existem diferentes tecnologias disponíveis, como as linguagens de programação Java, C, PHP, C#, C++, Delphi, entre tantas outras.

Além das linguagens de programação, existem os diferentes paradigmas, ou seja, diferentes formas de estruturar e executar programas de computador. Alguns dos principais paradigmas de programação são apresentados a seguir:

- **Imperativo:** paradigma associado a comandos sequenciais e a troca de estados de memória. Alguns exemplos de linguagens que permitem a programação no paradigma imperativo são: Cobol, C, Pascal, PHP, entre outras.
- **Orientado a objetos:** associação de comportamento a estruturas de dados chamados objetos. Exemplos de linguagens que permitem a programação orientada a objetos são: C++, Java, PHP, entre outras.
- **Concorrente**: trata-se da programação multitarefa, ou seja, o programa é criado para que mais de uma tarefa seja disparada simultaneamente, as quais são executados de forma concorrente, alternando o uso do processador. Java, C++ e Ruby são exemplos de linguagens que permitem a programação concorrente.
- **Lógico**: neste paradigma a computação é o resultado de funções lógicas, baseadas em fatos, regras e questões. A principal linguagem para programação lógica é o Prolog.
- **Funcional:** neste caso, a computação é o resultado de funções matemáticas, representadas através de listas. A principal linguagem utilizada para a programação neste paradigma é o Lisp.

Além desses paradigmas, algumas tendências atuais, na área de programação, são a programação orientada a serviços, programação paralela, programação segura, programação orientada a aspectos e programação ágil, entre outras.

5.2.4 Testes

A etapa de testes trata-se da execução de um sistema com o objetivo de encontrar falhas. O quadro 8 apresenta algumas definições importantes de serem diferenciadas sobre testes de software.

Quadro 8 – Definições de termos sobre teste de software

Termo	Definição
Teste	Processo de execução de um sistema, ou parte dele, com o objetivo de encontrar falhas.
Verificação	Teste para encontrar falhas em um sistema, em ambiente de testes ou simulado.
Validação	Teste para encontrar falhas em um sistema, em ambiente real.
Debug	Processo para descobrir a causa de uma falha encontrada.

Alguns dos princípios fundamentais da área de teste de software são apresentados a seguir:

- um bom caso de teste deve possuir alta probabilidade de encontrar erros e não de provar que o sistema funciona;
- um sistema não deve ser testado por quem o desenvolveu, visto que esse é o caso de menor probabilidade de se encontrar erros;
- todo o caso de teste deve especificar o resultado esperado, a fim de permitir que o testador saiba se o teste obteve sucesso ou não;
- deve-se criar casos de teste tanto para entradas válidas como para as inválidas;
- como testar todas as possibilidades de uso do sistema é praticamente impossível, devemos testar os casos mais importantes;
- deve-se cometer erros, propositalmente, para verificar o comportamento do sistema;

Entre os principais tipos de teste de software podemos citar o teste unitário, teste funcional, teste integrado, teste de regressão e o teste de sistema.

Teste unitário

O teste unitário, também chamado de caixa branca (do inglês *White Box*), tem como objetivo examinar a estrutura interna do sistema e, por isso, exige que o testador possua conhecimentos sobre a linguagem de programação utilizada na sua construção e sobre a estrutura interna do sistema.

O objetivo é garantir que cada caminho do programa seja executado pelo menos uma vez, a fim de garantir que não existem erros, os quais podem causar paradas no sistema.

Teste funcional

O teste funcional, também chamado de caixa preta (do inglês *Black Box*), é baseado

nos requisitos do sistema e não exige que o testador possua conhecimentos sobre a linguagem de programação ou sobre a estrutura interna do sistema. Entretanto, esse tipo de teste exige uma boa documentação dos requisitos do sistema e habilidade para determinar o número de testes adequados. Trata-se do teste das funcionalidades do sistema, via interface de usuário, a fim de verificar se o sistema funciona e se está de acordo com a especificação dos requisitos.

Teste integrado

Trata-se do processo de testar se as várias partes de um sistema funcionam, de forma integrada, sem gerar erros. Testa-se as interfaces entre as partes do sistema (módulos, componentes, subsistemas etc.).

Teste regressivo

Tem como objetivo detectar problemas causados por mudanças ou adição de novas funcionalidades em módulos já testados. O desafio, nesse tipo de testes, é selecionar os casos de teste que devem ser reexecutados a cada alteração ou adição de uma nova funcionalidade.

Teste de sistema

Tratam-se dos testes realizados no sistema depois que o mesmo está pronto. Para isso, os objetivos do sistema devem estar extremamente claros, indicando exatamente o que os usuários esperam do sistema. Necessita da interação entre a equipe de testes e alguns usuários.

Alguns tipos de teste de sistema são o teste de carga, de configuração, compatibilidade, segurança, performance, instalação, confiabilidade, recuperação, utilidade e usabilidade.

O quadro 9 apresenta um resumos dos principais tipos de teste de software.

Quadro 9 – Definições de termos sobre teste de software

Tipo de teste	**Objetivo**
Unitário	Testar a estrutura interna do sistema.
Funcional	Testar se o sistema funciona e se está em conformidades com a especificação.
Integrado	Testar se o sistema funciona após a junção de suas partes.
Regressivo	Testar se o sistema continua funcionando após mudanças ou adição de novas funcionalidades.

Sistema	Testar o sistema como um todo para verificar se está funcionando e de acordo com as necessidades dos usuários.

Antes de iniciarmos qualquer tipo de teste é necessário elaborar um plano de testes, no qual definimos as funcionalidades, regras e métodos a serem utilizados nos testes. Os principais elementos existentes em um plano de testes são: objetivos para cada fase de teste, prazos e responsáveis por cada atividade, padrões para a execução dos testes e registros dos resultados e os critérios para mensurar a realização e o sucesso dos testes.

É importante destacar que cada teste deve ser conduzido como um experimento, o qual deve ser cuidadosamente controlado e documentado, de tal forma que possa ser reproduzido.

As informações típicas de um relatório de resultado de testes são:

- nome do projeto;
- componentes e módulos testados;
- propósito dos testes;
- plano de testes;
- pessoas envolvidas;
- artifícios utilizados;
- configuração de hardware e software;
- resultados de cada teste, com os erros encontrados;
- assinatura do responsável pelos testes.

5.2.5 Implantação

Nesta etapa, todo o ambiente do cliente é preparado, o sistema é instalado e configurado e os usuários são treinados. Na implantação de um sistema, geralmente servidores de banco de dados e de aplicação são preparados e configurados para que o sistema possa ser utilizado. Além disso, são realizados os treinamentos que capacitam os usuários para a utilização do sistema. Essa etapa é encerrada somente quando tudo estiver pronto para que o sistema possa ser utilizado no cliente.

5.2.6 Manutenção

Trata-se de uma atividade que contempla a execução de todas as anteriormente descritas, para os casos em que um sistema já em uso tenha necessidade de alguma alteração, seja em função de uma correção ou de uma nova necessidade do cliente. As alterações para correção chamamos de manutenção corretiva e as alterações para

inclusão de novos serviços de manutenção evolutiva.

Em ambos os casos, a alteração do sistema deve ser cuidadosamente conduzida, a fim de evitar que alterações em uma parte afetem, indevidamente, outras partes.

Cada alteração deve ser conduzida como um miniprojeto, para o qual deve-se especificar os requisitos, adaptar o projeto do sistema, implementar e testar a alteração e as demais partes do sistema. Após os testes, deve-se gerar uma nova versão do sistema e implantar no cliente.

5.3 Atividades de apoio

5.3.1 Gerência de projetos

A gerência de projetos tem como objetivo principal realizar o planejamento e o controle de todas as atividades necessárias para a realização de um projeto, seja ele de software ou de outra área qualquer.

Projeto é a denominação dada a um empreendimento temporário com a finalidade de produzir um produto ou serviço único. O fato de um projeto ser temporário não implica ter curta duração, mas sim a existência de início e fim predeterminados (PMI, 2004). A gerência de projetos consiste na aplicação de conhecimentos, habilidades, ferramentas e técnicas às atividades do projeto, de forma a atingir as necessidades e expectativas das partes envolvidas.

Podemos definir um projeto como bem-sucedido quando ele foi desenvolvido e realizado:

- no prazo e no orçamento previstos;
- dentro das especificações técnicas e de qualidade previstas;
- cliente satisfeito com o produto ou serviço recebido.

Já os projetos malsucedidos apresentam problemas comuns, como, por exemplo:

- atrasos de cronograma;
- custos acima do previsto;
- falta de recursos humanos e materiais necessários;
- mudanças nos requisitos do projeto não controladas;
- qualidade do produto ou serviço abaixo da esperada;
- complexidade acima da capacidade da equipe;
- produtos mal projetados.

Para que um projeto atinja seus objetivos de forma eficaz, faz-se necessário um gerenciamento adequado. A ideia de gerenciamento de projetos iniciou com a definição de um guia genérico de boas práticas na gerência de projetos, o PMBOK (*Project*

Management Body of Knowledge), com padrões e definições capazes de guiar a gerência em qualquer tipo de projeto. A partir disso, o PMBOK foi se tornando o principal e mais adotado modelo de gerenciamento de projetos, na área de desenvolvimento de software (PMI, 2004).

O gerenciamento de um projeto é composto por processos, por mais simples que seja o contexto ao qual se aplica. Com isso, boas práticas para a execução destes processos são necessárias para que se tenha um controle sobre todo o andamento do projeto em questão. Os grupos de processos são cinco etapas pelas quais qualquer projeto passará: iniciação, planejamento, execução, controle e encerramento (PMI, 2008).

A ***iniciação*** do projeto compreende dois processos, em que é desenvolvido o termo de abertura e são definidas as partes interessadas do projeto. Com isso, o escopo inicial pode ser definido e os recursos financeiros são acordados. O grupo de iniciação é muito importante para se definir qual a necessidade e onde se quer chegar com o desenrolar do projeto.

O grupo de ***planejamento*** é aquele que engloba o maior número de processos. Isso se deve à necessidade de planejar muito bem o que será feito e como serão controladas as atividades que serão realizadas ao longo do projeto. É necessário definir com a maior precisão possível questões como escopo, tempo, custos e riscos do projeto, para que não ocorram surpresas e necessidades de replanejamento, já que o PMBOK sugere evitar, ao máximo, mudanças não previstas no planejamento.

O grupo de ***execução*** compreende processos que visam coordenar os recursos e atividades executadas, com o intuito de cumprir as necessidades definidas no projeto, em conformidade com o plano de gerenciamento realizado na etapa de planejamento. Este grupo compreende grande parte do orçamento do projeto e a execução dos processos pode evidenciar ou implicar mudanças, impactando em replanejamento, retrabalho e, possivelmente, atraso no projeto. Portanto, um gerenciamento eficaz da execução se faz necessário para o sucesso do projeto.

O grupo de processos de ***controle*** é composto por processos, os quais, como o nome já diz, definem práticas a serem executadas com o intuito de acompanhar e revisar o desempenho de projeto, validando se a execução de suas atividades está de acordo com o que foi planejado, monitorando recursos, mudanças e pontos críticos. São esses processos que proporcionam uma melhor visão de como está a saúde do projeto.

O grupo de processos de ***encerramento*** tem o intuito de finalizar o projeto, os quais definem boas práticas para se revisar o ciclo desenvolvido, obter aceitação do cliente, formalizar o término do projeto, encerrar as aquisições, documentar o desempenho do projeto e as lições aprendidas.

O quadro 10 apresenta um resumo dos grupos de processos da gerência de projetos.

Quadro 10 – Grupos de processo da gerência de projetos

Termo	Objetivo	Resultado
Iniciação	Definir o escopo inicial do projeto.	Termo de abertura
Planejamento	Planejar todas as atividades do projeto.	Plano do projeto
Execução	Executar as atividades de acordo com o planejado.	Projeto executado
Controle	Acompanhar e revisar o desempenho do projeto.	Relatório de desempenho
Encerramento	Obter a aceitação do cliente.	Aceite do projeto

A divisão do projeto por áreas de conhecimento é uma outra perspectiva de organização de processos, baseada no contexto em que se aplica o processo e, não mais na sua relação com o ciclo de vida do projeto. Os processos estão distribuídos em nove áreas de conhecimento: escopo, tempo, integração, riscos, aquisições, custos, qualidade, comunicação e recursos humanos (PMI, 2008).

O ***escopo*** é o conjunto de requisitos necessários para entregar o projeto e seus produtos de acordo com as expectativas do cliente e partes interessadas. Um projeto deve ser entregue exatamente conforme o escopo, sem nenhuma funcionalidade a mais, pois a entrega de funcionalidades não previstas requer tempo e esforço desnecessários e não contratados.

A gerência de ***tempo*** é sinônimo de garantia da conclusão do projeto dentro do prazo predeterminado. Se o projeto foi entregue no prazo acordado, significa que, de alguma forma, a gerência de tempo foi bem-sucedida. O gerenciamento do tempo do projeto inicia com a definição das atividades necessárias para o projeto, bem como com as estimativas de esforço e duração para cada atividade. Com o sequenciamento e as estimativas em mão, o gerente do projeto inicia a definição de um cronograma das atividades, definindo suas datas de início e fim, datas de entregas dos pacotes e marcos de controle do andamento do projeto.

A gerência de ***integração***, como o próprio nome já sugere, consiste no planejamento e no controle de todas as atividades que visam integrar as fases do projeto. Portanto, gerenciar a integração nada mais é do que planejar e controlar, em uma perspectiva macro, os processos que englobam os cinco grupos definidos pelo PMBOK.

O princípio do gerenciamento de ***riscos*** consiste em identificar problemas que podem afetar o desempenho do projeto, mas que ainda não aconteceram. Como já visto neste livro, um risco é algo que, quando ocorre, tem impacto negativo no projeto. Um projeto, independente de sua área de atuação, sempre está sujeito a riscos, podendo ser técnicos, de ambiente, de recursos, entre outros, cada um com sua prioridade. A gerência de riscos atua em cima disso, procurando identificar os possíveis riscos na etapa de planejamento, já associando medidas e formas de lidar

com os riscos que se concretizarem.

A gerência de ***aquisições*** dispõe de processos para garantir a aquisição e a administração de todos os recursos inerentes ao projeto e que não são supridos pela equipe do projeto. É uma área que atua diretamente com as necessidades já previstas para o projeto, planejando a aquisição como um todo, desde os fornecedores, critérios de avaliação de propostas, documentação contratual necessária para a aquisição dos recursos e encerramento das aquisições, quando este encerramento se aplicar.

O gerenciamento de ***custos*** envolve todos os processos necessários para a estimativa e o controle dos custos para que a realização do projeto ocorra de acordo com o planejado. De acordo com o PMI (2008), a gerência de custos dispõe de processos que visam garantir com que o projeto se encerre dentro do orçamento que foi previsto e aprovado pelo cliente. Gerenciar custos envolve três atividades básicas: estimar, orçar e controlar.

Gerenciar a ***qualidade*** é atuar com processos que auxiliam para que as funcionalidades e a utilização do produto final estejam em conformidade com o que foi especificado para o projeto. A ausência ou má aplicabilidade do gerenciamento da qualidade impacta diretamente nas funcionalidades do projeto, satisfação do cliente, necessidade de retrabalho, motivação da equipe, eficiência e eficácia do produto. O gerenciamento da qualidade se dá pelo planejamento, garantia e controle da qualidade.

A ***comunicação*** é fundamental em um projeto, pois é um trabalho em equipe e requer que a informação chegue a todos os interessados de forma harmônica e rápida. A gerência de comunicação remete bastante ao papel do gerente de projetos, que é o responsável por facilitar a distribuição do fluxo de informações dentro do projeto.

O gerenciamento de ***recursos humanos*** é liderado pela figura do gerente de projetos, que procura garantir que as pessoas possuem condições de realizar o trabalho para o qual estão designadas. A gerência de recursos humanos se inicia com a definição dos papéis, responsabilidades, posição hierárquica e grau de conhecimento das pessoas para as atividades pelas quais são responsáveis. É dever da gerência de recursos humanos planejar o desenvolvimento pessoal de cada integrante do projeto, necessidades de treinamento, *feedbacks*, premiações e aspectos da vida pessoal que possam interferir no andamento do projeto. Outro aspecto que é analisado na gerência de recursos humanos envolve a motivação das pessoas, o que está diretamente ligado com a satisfação para com o ambiente de trabalho.

O quadro 11 apresenta um resumo das nove áreas de conhecimento da gerência de projetos.

Quadro 11 – Áreas de conhecimento da gerência de projetos

Área de conhecimento	**Objetivo**
Escopo	Definir e controlar o escopo do projeto.
Tempo	Garantir a conclusão do projeto dentro do prazo, previamente planejado

	e acordado.
Integração	Planejar e controlar todas as atividades que visam integrar as fases do projeto.
Riscos	Identificar e monitorar os possíveis problemas do projeto.
Aquisições	Garantir a aquisição e administração de todos os recursos necessários ao projeto.
Custos	Estimar e controlar os custos para a realização do projeto.
Qualidade	Planejar, controlar e garantir a qualidade do projeto.
Comunicação	Garantir que a informação chegue a todos os interessados de forma correta e rápida.
Recursos Humanos	Definir, acompanhar e motivar a equipe do projeto para a realização de um bom trabalho.

5.3.2 Gerência da configuração

Durante um projeto de software, muitos documentos, diagramas, especificações e códigos-fonte são gerados e constantemente modificados. Assim, dependendo da complexidade do projeto e do tamanho da equipe, a grande quantidade de material de trabalho produzido e as diversas mudanças nele podem levar o projeto ao caos. É comum, nestes casos, a não localização do material necessário, a incerteza de qual é a versão mais atual, dentre tantas outras dificuldades que acarretam retrabalho e atraso nos projetos.

Por isso, a gerência de configuração de software tem como objetivo estabelecer e manter a integridade de todos os produtos de trabalho de um projeto e disponibilizá-los aos envolvidos de forma rápida e confiável (CHRISSIS, 2003).

O gerente de configuração é o responsável por saber responder o que mudou, quando mudou, porque mudou e quem fez as mudanças nos produtos de trabalho de um projeto. Por isso, o desafio para uma boa gerência de configuração é encontrar o equilíbrio entre o excesso de burocracia, necessária para manter a integridade de tudo, e o controle adequado das mudanças, de forma a evitar problemas.

A gerência de configuração de software envolve controle de versão, controle de mudanças e integração contínua.

Em relação ao ***controle de mudanças***, deve-se registrar toda e qualquer mudança, tendo como informações o solicitante, a justificativa e o responsável.

Já quanto ao ***controle de versão***, todo o artefato deve ser rastreável no que tange a alterações realizadas, sempre registrando-se quem, quando e onde.

Quanto à ***integração contínua***, o objetivo é garantir que o sistema evolua de forma estável e funcional, podendo retornar sempre a uma configuração anterior.

Para que a gerência de configuração seja conduzida de forma adequada, é necessária

a elaboração de um plano de gerência de configuração, no qual se estabelecem normas, ferramentas e modelos que permitem gerenciar, de maneira satisfatória, os itens de configuração de um sistema.

Um item de configuração é a designação geral de qualquer artefato definido para ser mantido sob gerência de configuração. Já configuração, é uma coleção de revisões de itens de configuração que reunidas formam um módulo, subsistema ou produto.

Em cada fase do desenvolvimento de um software, um conjunto bem definido de itens de configuração deve ser estabelecido. A este conjunto é dado o nome de linha de base (do inglês *Baseline*) ou configuração base do sistema. Uma linha de base é uma "fotografia" da configuração do software em um dado momento do tempo, normalmente estabelecida em momentos estratégicos do projeto. Uma vez fechada, a linha de base não pode mais ser alterada.

5.3.3 Gerência da qualidade

São objetivos da gerência da qualidade definir explicitamente os critérios de qualidade, estabelecer as atividades necessárias para garanti-la, executar essas atividades nos projetos, utilizar indicadores para monitorar e melhorar a qualidade e fornecer relatórios sobre o uso do processo.

É importante destacar que não é papel da gerência da qualidade executar testes de software ou inspeção de documentação e demais itens do projeto, mas sim garantir que o processo de teste, bem como todos os demais, sejam realizados conforme a definição de cada organização, em todos os projetos.

A gerência da qualidade reduz a quantidade de trabalho repetido, permite reduzir custos, poupa tempo e aumenta a qualidade do produto final.

Segundo Pressmann (2001), um software de qualidade está em concordância com os requisitos e com os padrões de desenvolvimento explicitamente definidos e com as características implícitas em todo software desenvolvido profissionalmente.

As principais atividades da gerência da qualidade são:

- estabelecimento de um plano de garantia da qualidade;
- participação na definição do processo de software;
- revisão das atividades de engenharia de software para o seu ajuste ao processo;
- garantir que os desvios sejam documentados e geridos segundo o procedimento estabelecido;
- registrar o que não estiver ajustado aos requisitos e reportar à gerência superior.

O quadro 12 apresenta um resumo das atividades fundamentais do processo de

software e o quadro 13 as atividades de apoio.

Quadro 12 – Atividades fundamentais do processo de software

Atividade	Objetivo	Resultado
Análise	Especificar as funcionalidades e restrições do sistema.	Especificação de requisitos do sistema
Projeto	Projetar a melhor estrutura para atender à especificação.	Diagramas técnicos
Implementação	Desenvolver, programar o sistema.	Sistema executável
Testes	Testar o sistema para identificação de falhas e não conformidades com a especificação.	Relatório de problemas
Implantação	Instalar e configurar o sistema, além de treinar os usuários.	Sistema pronto para o uso e usuários treinados
Manutenção	Corrigir defeitos, melhorar o sistema ou incluir novas funcionalidades.	Nova versão do sistema

Quadro 13 – Atividades de apoio do processo de software

Atividade	Objetivo	Resultado principal
Gerência de projeto	Planejar e controlar todas as atividades do projeto.	Plano de projeto e relatório de desempenho
Gerência de configuração	Controlar todos os artefatos do projeto.	Artefatos do projeto organizados e íntegros
Gerência de qualidade	Garantir que o processo definido está sendo seguido com qualidade.	Relatório de não conformidades

PARA COMPLEMENTAÇÃO DE ESTUDO

É importante destacar que este capítulo teve como objetivo apenas apresentar as principais atividades do processo de software, sem a pretensão de descrevê-las completamente. Para aprofundar seus estudos sobre o processo de software consulte livros sobre engenharia de software como, por exemplo, Sommerville (2005) e Pressmann (2001). Além disso, uma ótima referência sobre o processo de software é o livro *Managing The Software Process*, do autor Watts Humphrey (HUMPHREY,

2004).

CAPÍTULO 6

FERRAMENTAS DE APOIO AO PROCESSO DE SOFTWARE

Este capítulo apresenta as ferramentas de apoio ao processo de software, chamadas ferramentas CASE (do inglês *Computer-Aided Software Engineering*), que são softwares que apoiam o desenvolvimento de sistemas e o processo de evolução. Assim, são apresentados os objetivos e limitações dessas ferramentas, bem como suas diferentes possibilidades de classificação, além de alguns exemplos.

6.1 Objetivos e limitações

Através do uso de ferramentas CASE adequadas, pode-se otimizar os projetos de software, ou seja, é possível reduzir o tempo necessário para sua construção, diminuir os custos e aumentar a qualidade do produto final. Entretanto, como já discutido anteriormente neste livro, atualmente existem muitas ferramentas disponíveis e o desafio é saber escolher as ferramentas corretas. É preciso escolher a ferramenta certa para as características de cada organização como o tipo de projeto, tamanho da equipe, tecnologias utilizadas, dentre outras.

As ferramentas CASE incluem editores de texto, gerenciadores de requisitos, editores de diagramas, compiladores, depuradores, gerenciadores de cronograma, gerenciadores de versões, gerenciadores de defeitos, automatizadores de testes, entre outras.

Essas ferramentas proporcionam apoio ao processo de software pela automação de algumas atividades e pelo fornecimento de informações sobre o software que está sendo desenvolvido. Alguns exemplos de atividades que podem ser automatizadas são:

- diagramas técnicos, a partir do código-fonte;
- código-fonte parcial, a partir dos diagramas da etapa do projeto;
- testes, simulando um grande volume de usuários;
- depuração de programas.

Segundo Huff (HUFF, 1992), as melhorias obtidas nos projetos de software com o uso de ferramentas de apoio atingem a ordem de 40 por cento. Entretanto, as

melhorias possibilitadas por essas ferramentas são limitadas por dois fatores:

1. O desenvolvimento de software é uma atividade que tem como princípio básico o pensamento criativo e, por isso, depende essencialmente de intervenção humana;
2. O desenvolvimento de software é uma atividade, na grande maioria dos casos, realizada em equipe que exige muita interação entre as pessoas.

6.2 Classificação

As ferramentas CASE podem ser classificadas de diferentes formas, com o objetivo de auxiliar a compreensão sobre o seu papel em apoiar as atividades do processo de software. Sommerville (2005) propõe uma classificação em três diferentes perspectivas:

- perspectiva funcional: neste caso as ferramentas são classificadas de acordo com a sua função específica;
- perspectiva de processo: classificação de acordo com a atividade do processo que apoia;
- perspectiva de integração: classificadas de acordo com a maneira como são organizadas em unidades integradas, que fornecem apoio a uma ou mais atividades de processo.

O quadro 14 apresenta uma classificação de ferramentas CASE de acordo com a sua função. São listados diferentes tipos de ferramentas com exemplos específicos para cada tipo. Não se trata de uma lista completa, mas sim de um exemplo de classificação funcional.

Quadro 14 – Classificação funcional de ferramentas

Tipo de ferramenta	**Exemplos**
Planejamento	Ferramentas PERT, de estimativas, planilhas de cálculo
Edição	Editores de texto, editores de diagramas, processadores de texto
Gestão de mudanças	Acompanhamento de requisitos, sistemas de controle de mudanças
Gestão da configuração	Sistema de controle de versões, ferramentas de construção de sistemas
Prototipação	Linguagens de mais alto nível, geradores de interface com o usuário
Apoio ao método	Editores de projeto, dicionários de dados, geradores de código
Processamento de linguagem	Compiladores, interpretados
Análise de programas	Geradores de ref. cruzadas, analisadores estáticos, analisadores dinâmicos
Ferramentas de testes	Geradores de dados de testes, comparadores de arquivos

Debugging	Sistemas de *debugging* interativos
Documentação	Programas de *layout* de páginas, editores de imagens
Reengenharia	Sistemas de referências cruzadas, sistemas de reestruturação de programas

Fonte: Sommerville, 2005.

O quadro 15 apresenta uma classificação alternativa das ferramentas, sob a perspectiva de processo, ou seja, mostra as ferramentas utilizadas nas diferentes etapas do processo de software. Como pode ser observado na figura, as ferramentas para planejamento, edição de textos, preparação de documentação e gerenciamento de configuração são utilizadas durante todo o processo de desenvolvimento de software.

Quadro 15 – Classificação por processo

Ferramenta	Análise	Projeto	Implementação	Testes
Planejamento	X	X	X	X
Edição de textos	X	X	X	X
Documentação	X	X	X	X
Gerenciamento de mudança	X	X	X	X
Gerenciamento de configuração		X	X	
Prototipação	X			X
Apoio a métodos	X	X		
Processamento de linguagem		X	X	
Análise de programa			X	X
Depuração			X	X
Testes			X	X
Reengenharia			X	

Fonte: Sommerville, 2005.

A amplitude do apoio ao processo de software, proporcionado pela tecnologia CASE, é outra possível dimensão de classificação. Fuggeta (1993) propõe que os sistemas CASE sejam classificados em três categorias:

1. *ferramentas* que apoiam as tarefas de processos individuais;

2. *workbenches* que apoiam fases ou atividades de processo. Normalmente, consistem em um conjunto de ferramentas, com maior ou menor grau de integração;
3. *ambientes* que oferecem apoio a todo ou, pelo menos, a uma parte substancial do processo de software. Normalmente, incluem vários *workbenches* diferentes, que são integradas de alguma maneira.

A figura 23 ilustra a classificação proposta por Fuggeta e mostra alguns exemplos dessas classes de ferramentas.

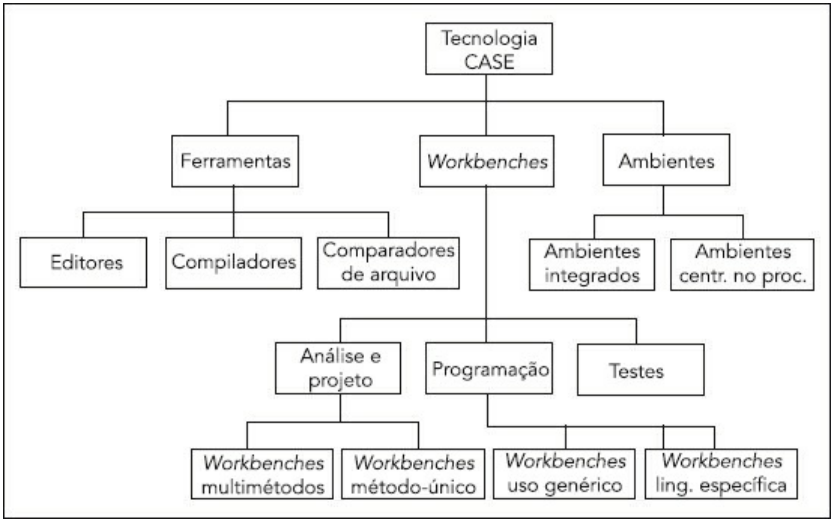

Figura 23 – Ferramentas, *Workbenches* e ambientes.
Fonte: Sommerville, 2005.

6.3 Exemplos

O quadro abaixo apresenta alguns exemplos de ferramentas CASE para as atividades fundamentais do processo de software, bem como o quadro 17 apresenta exemplos de ferramentas para as atividades de apoio.

Quadro 16 – Exemplos de ferramentas CASE para as atividades fundamentais

Atividade	Ferramenta	Fabricante	Site
Análise	DOORS Requisite Pro CliberRM	IBM IBM Borland	www.ibm.com/software/awdtools/doors www.ibm.com/software/awdtools/reqpro www.borland.com/br/products/caliber/rm.html
Projeto	Astah Argo UML Enterprise Architect Rational Rose	Change Vision Tigres.org Sparx Symtens IBM	http://astah.change-vision.com http://argouml.tigres.org www.sparxsystems.com www.ibm.com/software/awdtools/developer/rose
Implementação	Eclipse Visual Studio NetBeans	Eclipse Foundation Microsoft NetBeans Community	www.eclipse.org www.microsoft.com/visualstudio http://netbeans.org
Testes	Bugzila Quality Center Silk	Mozilla Foundation HP Borland	www.bugzilla.org www.hp.com/country/us/en/prodserv/software.html www.borland.com/us/products/silk

Quadro 17 – Exemplos de ferramentas CASE para as atividades de apoio

Atividade	Ferramenta	Fabricante	Site
Gerência de projeto	Project JIRA Project-open	Microsoft Atlassian Project-open	www.microsoft.com/project/ www.atlassian.com/software/jira/ www.project-open.com/
Gerência de configuração	Subversion CVS ClearQuest	Tigris GNU Project IBM	http://subversion.tigris.org/ www.nongnu.org/cvs/ www.ibm.com/software/awdtools/clearquest
Gerência da qualidade	Quality Center	HP	www.hp.com/country/us/en/prodserv/software.html

PARA COMPLEMENTAÇÃO DE ESTUDO

Atualmente, existem muitas ferramentas CASE disponíveis no mercado, tanto comerciais como gratuitas. Para complementar seu estudo sofre as ferramentas de apoio ao processo de software consulte as referências Fuggeta (1993) e Sommerville (2005), além dos sites dos principais fabricantes listados nos quadros 16 e 17 deste

capítulo.

CAPÍTULO 7

MODELOS DE PROCESSO DE SOFTWARE

Este capítulo apresenta alguns modelos do processo de software, que são diferentes formas de execução do processo descrito no Capítulo 5. Além dos modelos genéricos, são apresentados brevemente o *Rational Unified Process* (RUP) e o *eXtreme Programming* (XP).

Um modelo de processo de software é uma representação abstrata de um processo de software. Cada modelo de processo representa um processo, a partir de uma perspectiva particular (SOMMERVILLE, 2005).

7.1 Modelos genéricos

Os modelos de processo genéricos não são descrições definitivas do processo de software. Em vez disso, são abstrações úteis, que podem ser utilizadas para explicar diferentes abordagens do desenvolvimento de software. Para muitos sistemas de grande porte, naturalmente, não existe apenas um processo de software que possa ser utilizado. Processos diferentes são utilizados para desenvolver diferentes partes do sistema (SOMMERVILLE, 2005).

7.1.1 Cascata

Este modelo sugere uma abordagem sistemática e sequencial para o desenvolvimento de software, e considera as atividades de especificação, desenvolvimento, validação e evolução, que são fundamentais ao processo, e as representa como fases separadas do processo, como a especificação de requisitos, o projeto de software, a implementação, os testes e assim por diante (PRESSMAN, 2001) e (SOMMERVILLE, 2005).

O modelo original em cascata prevê "ciclos de realimentação", como pode ser visto na figura 24. Porém, a grande maioria das organizações que aplica esse modelo de processo o trata como sendo estritamente linear.

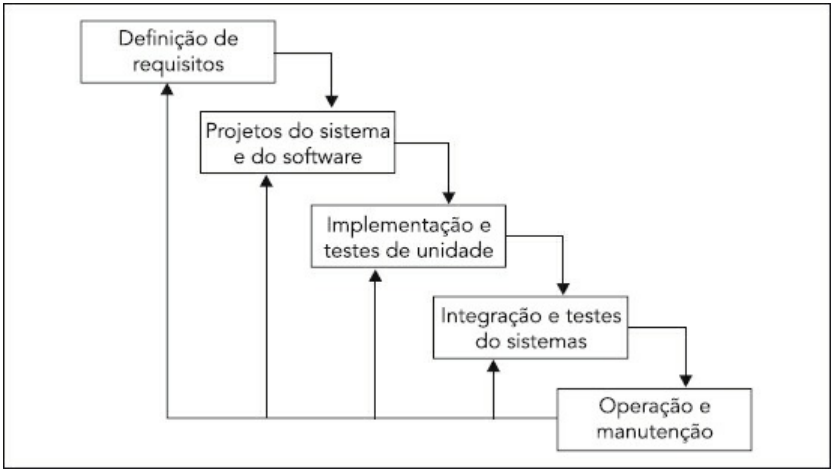

Figura 24 – O modelo original em cascata.
Fonte: Sommerville, 2005.

O principal problema com o modelo em cascata é sua inflexível divisão do projeto nessas fases distintas. Portanto, o modelo em cascata deve ser utilizado somente quando os requisitos forem bem compreendidos, na sua totalidade.

7.1.2 Evolucionário

O desenvolvimento evolucionário tem como princípio o desenvolvimento de uma implementação inicial e, posteriormente, após a exposição do resultado ao comentário do usuário, fazer seu aprimoramento por meio de muitas versões, até que um sistema adequado tenha sido desenvolvido (Figura 25).

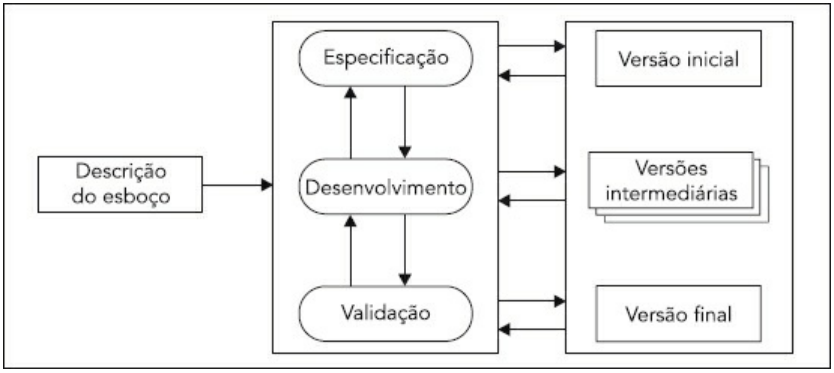

Figura 25 – Desenvolvimento evolucionário.
Fonte: Sommerville, 2005.

Há dois tipos de desenvolvimento evolucionário: exploratório e prototipação descartável. No caso do desenvolvimento exploratório, o objetivo é trabalhar com o cliente, a fim de explorar seus requisitos e entregar um sistema final que evolui a partir de uma especificação genérica inicial. Já na prototipação descartável, o objetivo é fazer experimentos com as partes dos requisitos que estejam mal compreendidas, através de protótipos.

7.1.3 Formal

O desenvolvimento formal de sistemas, ilustrado na figura 26, tem semelhanças com o modelo em cascata, porém é uma abordagem que se baseia na produção de uma especificação formal matemática do sistema e na transformação dessa especificação, utilizando-se métodos matemáticos, para construir um programa. As transformações preservam a conformidade, de tal forma que possa ser diretamente mostrado que o programa está de acordo com a sua especificação.

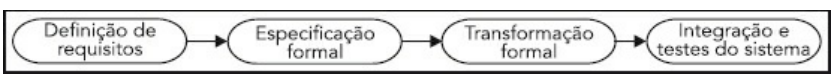

Figura 26 – Desenvolvimento formal de sistemas.
Fonte: Sommerville, 2005.

Esse modelo é indicado para o desenvolvimento de sistemas críticos, especialmente

aqueles em que um estudo de segurança deve ser feito antes de pôr o sistema em operação.

Contudo, esse é um modelo de processo que exige habilidades especiais para sua execução, além de dificultar a especificação de alguns aspectos como, por exemplo, a interface.

7.1.4 Orientado a reuso

O desenvolvimento orientado a reuso é uma abordagem que tem como base a existência de um número significativo de componentes reutilizáveis. O processo de desenvolvimento de sistemas se concentra na integração desses componentes, em vez de proceder ao desenvolvimento a partir do zero.

Segundo Sommerville (2005), existem quatro estágios nesta abordagem, conforme ilustra a figura 27:

- *análise de componentes* – considerando a especificação de requisitos, é feita uma busca de componentes para implementar essa especificação;
- *modificação de requisitos* – durante esse estágio os requisitos são analisados, utilizando-se as informações sobre os componentes já existentes; caso seja possível, alguns requisitos são modificados para refletir os componentes disponíveis;
- *projeto de sistema com reuso* – durante essa fase, a infraestrutura do sistema é projetada ou reutilizada de projetos anteriores;
- *desenvolvimento e integração* – os componentes faltantes são desenvolvidos e, após, é realizada a integração de todos os componentes do sistema.

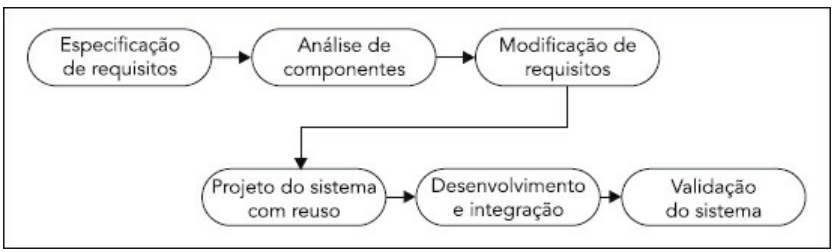

Figura 27. Desenvolvimento orientado a reuso.
Fonte: Sommerville, 2005.

7.1.5 Incremental

O desenvolvimento incremental é uma abordagem intermediária entre o modelo em cascata e o desenvolvimento evolucionário, que combina vantagens de ambos.

Em um processo de desenvolvimento incremental (Figura 28), os clientes identificam, em um esboço, as funções a serem fornecidas pelo sistema. Após, definem a prioridades dessas funções. Em seguida, é definida uma série de estágios de entrega, com cada estágio fornecendo um subconjunto de funcionalidades do sistema. A alocação de funções aos estágios depende da prioridade da função, e as de maior prioridade são entregues primeiramente ao cliente.

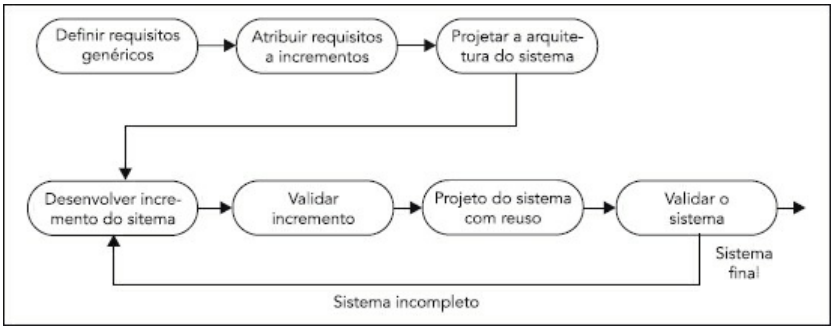

Figura 28 – Desenvolvimento incremental.
Fonte: Sommerville, 2005.

Neste caso, é possível utilizar diferentes modelos para o desenvolvimento de cada incremento. Por exemplo, quando as funções em um incremento têm uma especificação bem-definida, o modelo de desenvolvimento em cascata pode ser utilizado. Já quando a especificação for mal definida, poderá ser utilizado um modelo de desenvolvimento evolucionário.

Através do modelo incremental, os clientes não precisam esperar até que todo o sistema seja entregue, para então tirarem proveito dele. Além disso, podem utilizar os primeiros incrementos como um protótipo e obter uma experiência que forneça os requisitos para estágios posteriores do sistema. Com isso, o risco de fracasso do projeto diminui consideravelmente. Uma outra característica do modelo incremental é o fato de que as funções prioritárias, que são entregues nos primeiros incrementos, passam por um maior número de testes.

7.1.6 Espiral

No desenvolvimento em espiral (Figura 29), o processo é representado como uma espiral em vez de uma sequência de atividades com caminhos de retorno. Cada volta na espiral representa uma fase no processo e não há fases fixas, tais como especificação ou projeto. Neste caso, os riscos são explicitamente avaliados e resolvidos durante todo o processo.

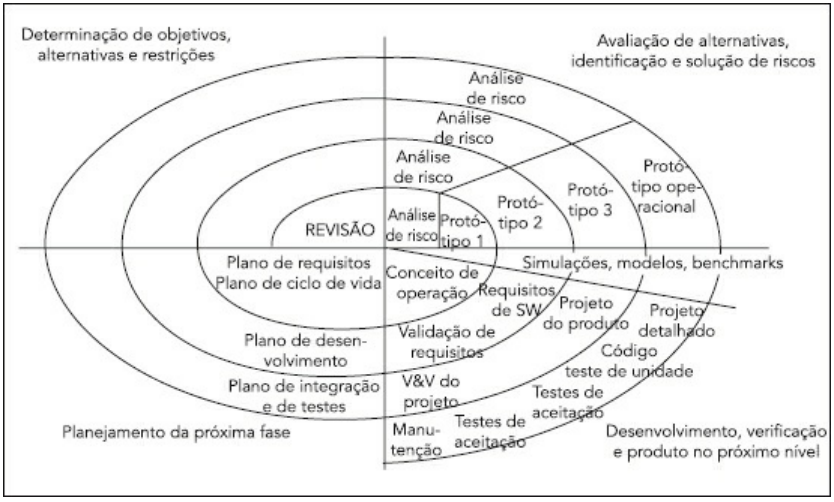

Figura 29 – Desenvolvimento em espiral.
Fonte: Boehm, 1988.

Conforme ilustrado na figura 29, cada volta da espiral é dividida em quatro setores:

1. *Definição de objetivos* – São definidos os objetivos específicos para essa fase do projeto. São identificadas as restrições para o processo e o produto, e é preparado um plano de gerenciamento detalhado. São identificados os riscos do projeto e, dependendo dos riscos, poderão ser planejadas estratégias alternativas;
2. *Avaliação de redução de riscos* – Para cada um dos riscos de projeto, é realizada uma análise detalhada e são tomadas providências para reduzir esses riscos;
3. *Desenvolvimento e validação* – Depois da avaliação dos riscos, é escolhido um modelo de desenvolvimento para o sistema;

4. *Planejamento* – O projeto é revisto e é tomada uma decisão sobre continuar com a próxima volta da espiral. Se a decisão for de continuar, serão traçados os planos para a próxima fase do projeto.

O quadro 18 apresenta um resumo dos modelos genéricos do processo de software.

Quadro 18 – Modelos genéricos do processo de software

Modelo	**Característica principal**
Cascata	Sequência em cascata de uma fase para outra. A fase seguinte não deve iniciar até que a fase precedente tenha sido concluída.
Evolucionário	Evolução de um sistema final a partir de uma especificação genérica inicial. O desenvolvimento inicia com as partes do sistema que estão melhor compreendidas.
Formal	Baseia-se na transformação de uma especificação matemática, por meio de diferentes representações, para um programa executável. Muito utilizando no desenvolvimento de sistemas críticos.
Baseado em reuso	Baseia-se no reuso sistemático, em que sistemas são integrados a partir de componentes já existentes. Possui quatro estágios: análise dos componentes, modificação de requisitos, projeto do sistema com reuso e desenvolvimento/integração.
Incremental	Quebra o desenvolvimento e a entrega em incrementos, com cada incremento entregando parte da funcionalidade requerida. O desenvolvimento inicia com os requisitos de maior prioridade para o cliente e, uma vez que um incremento é iniciado, os requisitos são congelados, ainda que os requisitos para incrementos posteriores continuem a evoluir.
Espiral	O processo é representado como uma espiral, em vez de uma sequência de atividades com caminhos de retorno. Cada volta na espiral representa uma fase no processo e não há fases fixas, tais como especificação ou projeto.

7.2 *Rational Unified Process* (RUP)

O RUP trata-se de um modelo de processo de software que fornece uma forma organizada para atribuir tarefas e responsabilidades dentro de uma empresa de desenvolvimento de software. Seu objetivo é assegurar a produção de software de alta qualidade, que atenda às necessidades de seus usuários finais, com prazos e orçamentos previsíveis (KRUCHTEN, 2003).

A figura 30 apresenta a arquitetura geral do RUP, em que o eixo horizontal representa o tempo e mostra as fases do ciclo de vida do processo à medida que se desenvolve, e o eixo vertical representa os processos (disciplinas) que agrupam as atividades de maneira lógica e evidenciam a variação da ênfase através do tempo.

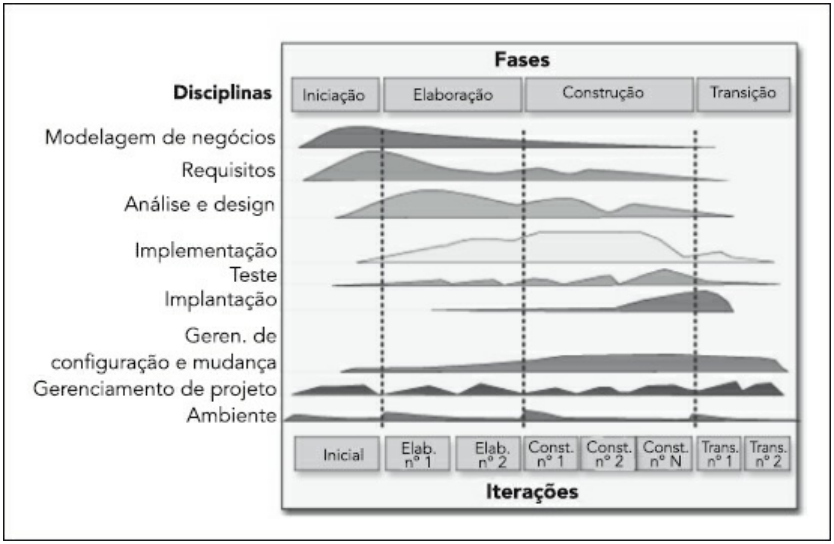

Figura 30 – Arquitetura geral RUP.
Fonte: Kruchten, 2003.

A partir de uma perspectiva de gerenciamento, o ciclo de vida de software do RUP é dividido em quatro fases sequenciais, cada uma concluída por um marco principal, ou seja, cada fase é basicamente um intervalo de tempo entre dois marcos principais. Em cada final de fase é executada uma avaliação para determinar se os objetivos da fase foram alcançados. Uma avaliação satisfatória permite que o projeto passe para a próxima fase.

O RUP captura, de forma satisfatória, muitas das melhores práticas utilizadas no desenvolvimento moderno de software permitindo, assim, seu uso em diversos projetos e organizações (KRUCHTEN, 2003).

7.3 *eXtreme Programming* (XP)

Extreme Programming é uma metodologia ágil, voltada para equipes pequenas ou médias que trabalham com requisitos vagos ou que mudam frequentemente. Segundo Teles (2004), a metodologia parte do princípio de que o cliente aprende sobre suas necessidades à medida que manipula o sistema que está sendo produzido. De acordo com Beck (2004), *eXtreme Programming* é baseada em alguns princípios de senso

comum, dentre eles:

- constante revisão de código através da programação em pares;
- constantes testes de unidade. O desenvolvimento é dirigido por testes;
- constante design (*refactoring*);
- simplicidade no projeto do sistema;
- constante refinamento de arquitetura;
- constantes testes de integração;
- iterações curtas.

Essa metodologia se diferencia de outras tradicionais por fornecer *feedback* contínuo através de curtos ciclos de iteração, ter uma abordagem incremental, tornar o cronograma flexível, respondendo às mudanças nas regras de negócio, confiança em testes automatizados, comunicação oral, *refactoring* e a colaboração entre os desenvolvedores (BECK, 2004).

PARA COMPLEMENTAÇÃO DE ESTUDO

Utilize as referências Sommerville (2005), Kruchten (2003) e Beck (2004) para complementar o seu estudo sobre os modelos de processo de software, os quais foram brevemente apresentados neste capítulo.

CAPÍTULO 8

MODELOS DE MELHORIA DE PROCESSOS DE SOFTWARE

Desde a década de 1980, iniciaram-se esforços para melhoria de processos de software, com o objetivo de melhorar a qualidade, aumentar a produtividade e reduzir os custos. Assim, este capítulo apresenta os principais modelos para melhoria do processo de software, definidos pelas normas ISO/IEC 12207 e 15504, além do modelo internacional *Capability Maturity Model Integration* (CMMI) e modelo nacional chamado de Melhoria de Processo do Software Brasileiro (MPS.BR).

As normas e os modelos de qualidade de processos têm como principal objetivo apoiar a definição e a melhoria dos processos de software nas organizações.

Normas internacionais de qualidade são criadas pelo trabalho voluntário de especialistas no mundo todo, sendo que estas normas tornaram-se a base para especificar produtos e organizar o fornecimento de serviços. São criadas de maneira formal, regulamentada, escritas seguindo regulamentos e aprovadas por instituições reconhecidas publicamente como capacitadas para tal.

Os organismos internacionais mais importantes para o setor de software são: a ISO (*International Organization for Standardization*), organização não governamental, estabelecida em 1947, com a missão de promover o desenvolvimento da normatização e atividades relacionadas a nível mundial, e a IEC (*International Electrotechnical Commission)*, fundada em 1906, organização mundial que publica as normas internacionais relacionadas com eletricidade, eletrônica e áreas relacionadas.

As principais normas para processos de software são: ISO/IEC 12207:2008 – *Software Life Cycle Processes, framework* para processos de ciclo de vida, e ISO/IEC 15504:2003 – *framework* para avaliação e melhoria de processos.

Diferentes modelos são utilizados, dependendo do mercado-alvo das organizações de software e do seu planejamento estratégico. Os modelos de referência na melhoria de processos de software mais difundidos, atualmente, são o CMMI (*Capability Maturity Model Integration*) e o MPS.BR (Melhoria de Processo do Software Brasileiro).

8.1 ISO/IEC 12207

A ISO publicou algumas normas para a área de software, dentre as quais a ISO/IEC 12207, que tem como objetivo apoiar a melhoria dos processos de software.

O objetivo principal desta norma é estabelecer uma estrutura comum para os processos do ciclo de vida de software, visando ajudar as organizações a compreenderem todos os componentes presentes na aquisição e no fornecimento de software e, assim, conseguirem firmar contratos e executarem projetos de forma mais eficaz (ISO/IEC 12207:2008).

O modelo visa à avaliação da empresa nos requerimentos do sistema de gestão da qualidade e abrange todo o processo de desenvolvimento de produto, produção, gestão de processos e melhoria contínua.

Como pode ser observado na figura 31, a norma ISO/IEC 12207 é agrupada em cinco processos fundamentais, oito processos de apoio e quatro processos organizacionais.

Os processos fundamentais dividem-se em Aquisição, Fornecimento, Desenvolvimento, Operação e Manutenção. Os processos de apoio dividem-se em Documentação, Gerência de Configuração, Garantia da Qualidade, Verificação, Validação, Revisão Conjunta, Auditoria e Resolução de Problema. Por fim, os processos organizacionais dividem-se em Gerência, Melhoria, Infraestrutura e Treinamento.

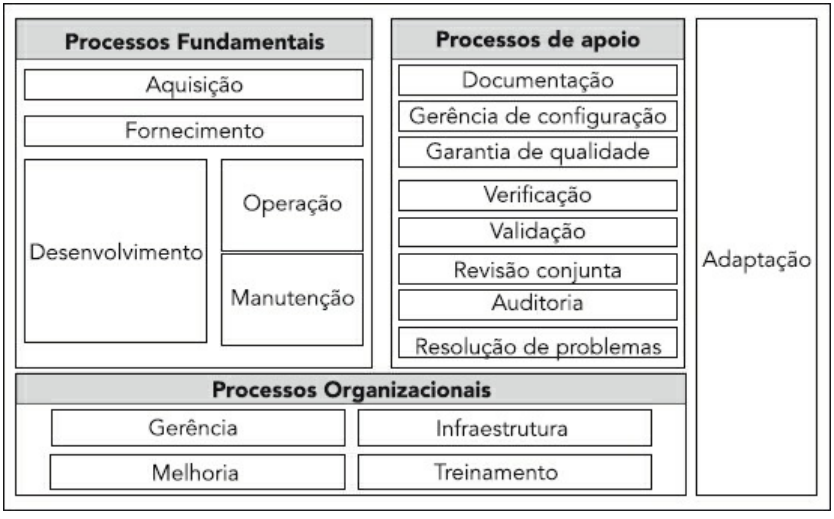

Figura 31 – Processos do ciclo de vida do software.
Fonte: Rocha *et al*., 2004.

8.2 ISO/IEC 15504

A norma ISO/IEC 15504 é também conhecida como SPICE (*Software Process Improvement and Capability dEtermination*) e sua utilização se destina à realização de avaliações de processos de software com dois objetivos: a melhoria dos processos e a determinação da capacidade de processos de uma organização (ISO/IEC 15504:2008).

Se o objetivo for a melhoria dos processos, a organização pode realizar a avaliação gerando um perfil dos processos que serão usados para a elaboração de um plano de melhorias, conforme ilustra a figura 32. Nesse caso, a organização deve definir os objetivos e o contexto, bem como escolher o modelo e o método para a avaliação e definir os objetivos de melhoria (ROCHA *et al.*, 2004).

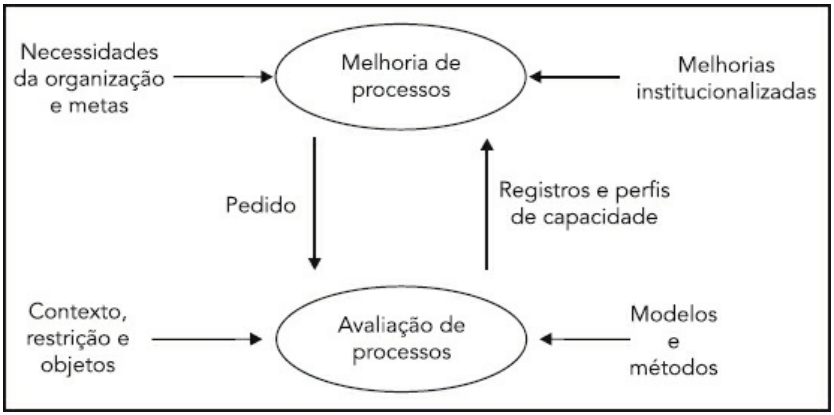

Figura 32 – Melhoria de processos ISO/IEC 15504.
Fonte: Rocha *et al.*, 2004.

No segundo caso, a empresa tem o objetivo de avaliar um fornecedor em potencial, obtendo o seu perfil de capacidade, conforme ilustra a figura 33. Para isso, a empresa deve definir os objetivos e o contexto da avaliação, os modelos e métodos de avaliação e os requisitos esperados. O perfil de capacidade permite ao contratante estimar o risco associado à contratação daquele fornecedor em potencial, para auxiliar na tomada de decisão de contratá-lo ou não (ROCHA *et al.*, 2001).

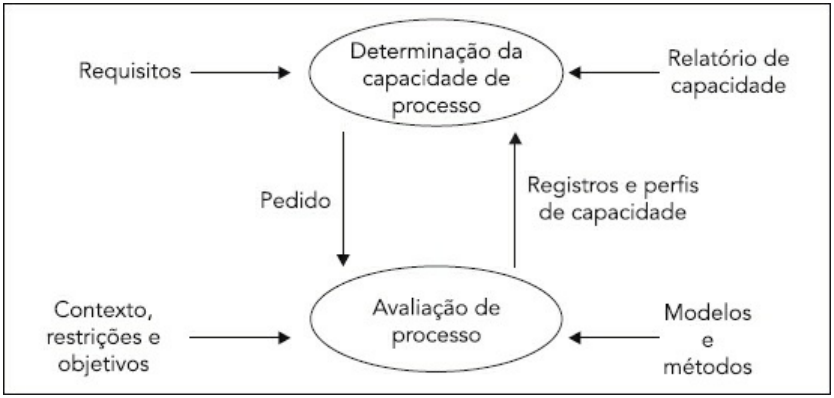

Figura 33 – Avaliação da capacidade ISO/IEC 15504.
Fonte: Rocha *et al*., 2004.

O modelo de referência estabelecido na norma ISO/IEC 15504 define duas dimensões: a dimensão do processo, que corresponde à definição de um conjunto de processos considerados universais e fundamentais para a boa prática da engenharia de software, e a dimensão de capacidade, que corresponde à definição de um modelo de medição com base na identificação de um conjunto de atributos que permite determinar a capacidade de um processo para atingir seus propósitos, gerando os produtos de trabalho e os resultados estabelecidos (ISO/IEC 15504:2008).

8.3 CMMI

O objetivo do modelo integrado de maturidade de capacidade (*Capability Maturity Model Integration* – CMMI) é servir de guia para a melhoria de processos nas organizações e, também, da habilidade dos profissionais em gerenciar o desenvolvimento, aquisição e manutenção de produtos ou serviços (CHRISSIS, 2003).

Sua primeira versão foi publicada em agosto de 2000 pelo *Software Engeneering Institute* (SEI) e, desde então, muitas organizações de software, no mundo todo, têm aderido a esse modelo de melhoria.

O CMMI possui duas representações: contínua e por estágios. A representação contínua possibilita à organização utilizar a ordem de melhoria que melhor atende aos objetivos de negócio da empresa e é caracterizado por níveis de capacidade, sendo eles: incompleto, executado, gerenciado, definido, quantitativamente gerenciado ou em otimização. Já a representação por estágios disponibiliza uma sequência

predeterminada para melhoria baseada em estágios, em que cada estágio serve de base para o próximo e é caracterizado por cinco níveis de maturidade: inicial, gerenciado, definido, qualitativamente gerenciado e em otimização (SEI, 2010).

O quadro 19 apresenta os cinco níveis do CMMI.

Quadro 19 – Níveis de maturidade do CMMI

Nível 1	**Inicial**	Os processos são caóticos. Geralmente a organização não possui um ambiente estável de desenvolvimento de software.
Nível 2	**Gerenciado**	Os projetos da organização possuem requisitos gerenciados e processos planejados, medidos e controlados. Dessa forma, eventuais não conformidades são identificadas com antecedência.
Nível 3	**Definido**	Os processos são bem caracterizados e entendidos, através do uso de padrões, procedimentos, ferramentas e métodos bem definidos. Esses fatores diferem o nível 3 do nível 2.
Nível 4	**Gerenciado quantitativamente**	Aumento da previsibilidade do desempenho de processos, graças ao controle quantitativo.
Nível 5	**Otimizado**	Os processos são continuamente melhorados com base em um entendimento quantitativo das causas comuns de alterações de desempenho. A melhoria contínua é obtida com inovações e melhor uso de tecnologias.

A figura 34 apresenta a estrutura e os componentes do modelo CMMI, representação por estágios e níveis de maturidade.

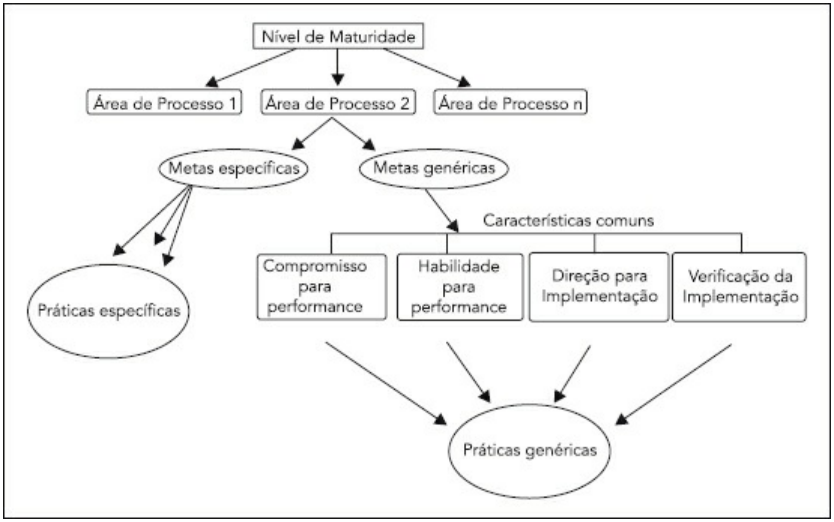

Figura 34 – Componentes do modelo CMMI.
Fonte: Chrissis, 2003.

Como pode ser observado na figura, o CMMI é formando pelos seguintes componentes:

- níveis de maturidade (*Maturity Levels*);
- áreas de processo (*Process Areas – PAs*);
- metas genéricas (*Generic Goals – GG*);
- metas específicas (*Specific Goals – SG*);
- práticas genéricas (*Generic Practices – GP*);
- práticas específicas (*Specific Practices – SP*).

Cada área de processo é composta por práticas específicas e genéricas, que quando implementadas irão satisfazer as metas da área de processo, que por fim irão resultar em um nível de maturidade para a organização.

As metas específicas são aplicadas a uma área de processo e descrevem as características específicas que devem ser implementadas para satisfazer plenamente a área. As metas específicas são itens obrigatórios e a ausência da sua implementação resultará em uma não satisfação da área. Já as práticas específicas são atividades relevantes que irão auxiliar no cumprimento das metas específicas. As práticas não são componentes obrigatórios, mas são itens esperados pelo modelo.

As metas genéricas são assim designadas, pois são encontradas em diversas áreas

de processos. O cumprimento das metas de uma área de processo irá representar um melhor planejamento e implementação dos processos daquela área. As metas genéricas são componentes obrigatórios no modelo e são determinantes na avaliação das áreas de processo. As práticas genéricas são características comuns que devem estar presentes em todas as áreas de processo e que irão garantir a institucionalização do processo. Não são itens obrigatórios, mas são itens esperados pelo modelo que irão auxiliar no cumprimento das metas genéricas.

O nível de maturidade é o resultado do cumprimento de todas as metas específicas e genéricas de todas as áreas de processo contempladas pelo nível. Cada nível de maturidade é responsável por estabilizar uma determinada parte dos processos da organização. As áreas contempladas em cada um dos níveis de maturidade são apresentadas no quadro 20.

Quadro 20 – Áreas dos níveis de maturidade do CMMI

Nível 1	**Inicial**	**Nenhuma**
Nível 2	**Gerenciado**	➜ Medição e Análise ➜ Gerência de Configuração ➜ Garantia da Qualidade de Processo e Produto ➜ Gerência de Acordo com Fornecedores ➜ Monitoramento e Controle de Projeto ➜ Planejamento de Projeto ➜ Gerência de Requisitos
Nível 3	**Definido**	➜ Análise de Decisão ➜ Gerência de Riscos ➜ Gerência Integrada de Projeto ➜ Treinamento Organizacional ➜ Definição do Processo Organizacional ➜ Foco no Processo Organizacional ➜ Validação ➜ Verificação ➜ Integração de Produto ➜ Solução Técnica ➜ Desenvolvimento de Requisitos
Nível 4	**Gerenciado quantitativamente**	➜ Desempenho do Processo Organizacional ➜ Gerência Quantitativa de Projeto
Nível 5	**Otimizado**	➜ Inovação Organizacional ➜ Análise de Causas e Resolução de Problemas

8.4 MPB.BR

O MPS.BR é um programa mobilizador, de longo prazo, criado em dezembro de 2003, coordenado pela Associação para Promoção da Excelência do Software Brasileiro (SOFTEX), que conta com apoio do Ministério da Ciência e Tecnologia (MCT), Financiadora de Estudos e Projetos (FINEP), Serviço Brasileiro de Apoio às Micro e Pequenas Empresas (SEBRAE) e Banco Interamericano de Desenvolvimento (BID) (SOFTEX, 2009).

O MPS.BR é um modelo para avaliação e melhoria de processos de software, que foi desenvolvido com base técnica nas normas ISO/IEC 12207 e suas emendas 1 e 2, e a ISO/IEC 15504, além de ser compatível com todo o conteúdo presente no modelo CMMI.

Os principais componentes do MPS.BR, conforme ilustrado na figura 35, são:

- Modelo de Referência (MR-MPS): contém os requisitos a serem atendidos pelas organizações, bem como os processos e os níveis de maturidade;
- Método de Avaliação (MA-MPS): contém o processo de avaliação, os requisitos para os avaliadores e os requisitos para averiguação da conformidade ao MR-MPS;
- « Modelo de Negócio (MN-MPS): contém as regras para implantação do MPS.BR pelas empresas de consultoria, de software e avaliação.

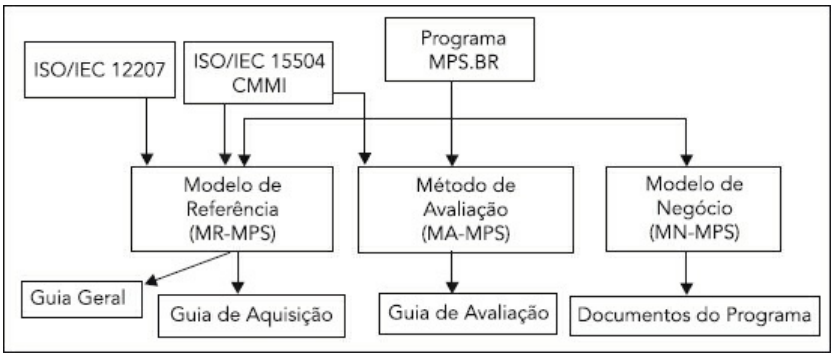

Figura 35 – Componentes do modelo MPS.BR.
Fonte: SOFTEX, 2009.

Além disso, todo o conteúdo do MPS.BR está organizado e documentado através de guias.

- Guia Geral: contém a descrição geral do MPS.BR e detalha o Modelo de Referência (MR-MPS), seus componentes e as definições comuns necessárias para seu entendimento e aplicação;

- Guia de Aquisição: descreve um processo de aquisição de software e serviços correlatos. É descrito como forma de apoiar as instituições que queiram adquirir produtos de software e serviços correlatos apoiando-se no MR-MPS;
- Guia de Avaliação: descreve o processo e o método de avaliação MA-MPS, os requisitos para avaliadores líderes, avaliadores adjuntos e Instituições Avaliadoras;
- Guia de Implementação: composto de sete partes, cada uma delas descrevendo como implementar um determinado nível do MR-MPS.

Os níveis de maturidade representam os estágios da evolução do processo da organização, sendo que a cada nível atingido a empresa estará evoluindo um conjunto de processos. No MPS.BR existem sete níveis de maturidade que combinam processos e capacidades de processos. Os níveis de maturidade do MPS.BR são os seguintes: A (Em Otimização), B (Gerenciado Quantitativamente), C (Definido), D (Largamente Definido), E (Parcialmente Definido), F (Gerenciado) e G (Parcialmente Gerenciado). A escala de maturidade se inicia no nível G e progride até o nível A.

O quadro 21 apresenta os processos pertencentes a cada um dos níveis de maturidade do MPS.BR.

Quadro 21 – Processos dos níveis de maturidade do MPS.BR

Nível	**Processos**
A	Análise de Causas de Problemas e Resolução – ACP
B	Gerência de Projetos – GPR (evolução)
C	Gerência de Riscos – GRI Desenvolvimento para Reutilização – DRU Análise de Decisão e Resolução – ADR Gerência de Reutilização – GRU (evolução)
D	Verificação – VER Validação – VAL Projeto e Construção do Produto – PCP Integração do Produto – ITP Desenvolvimento de Requisitos – DRE
E	Gerência de Projetos – GPR (evolução) Gerência de Reutilização – GRU Gerência de Recursos Humanos – GRH Definição do Processo Organizacional – DFP Avaliação e Melhoria do Processo Organizacional – AMP
F	Medição – MED Garantia da Qualidade – GQA Gerência de Configuração – GCO Aquisição – AQU
G	Gerência de Requisitos – GRE

A figura 36 apresenta uma comparação dos níveis de maturidade do MBR.BR e CMMI.

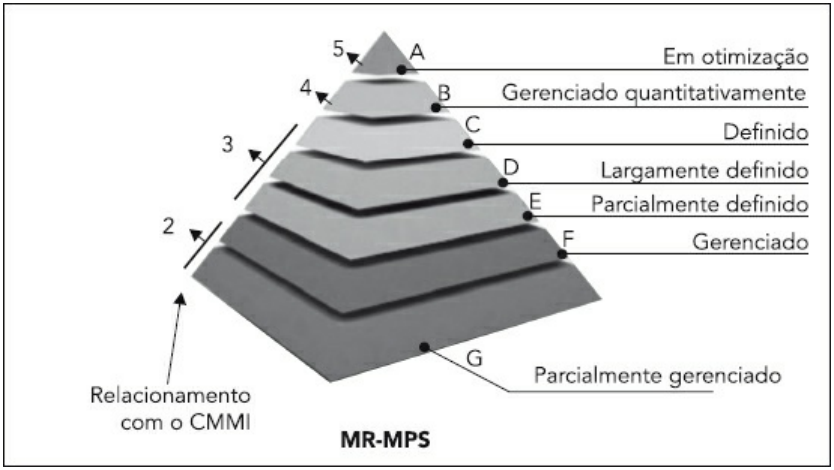

Figura 36 – Relação entre os níveis de maturidade do MPS.BR e CMMI.
Fonte: SOFTEX, 2009.

A capacidade do processo representa um conjunto de atributos de processo que devem ser atingidos por ele. Ela estabelece o quão detalhada, organizada e institucionalizada está a execução do processo na organização. No MPS.BR existem nove atributos de processo, que irão definir a sua capacidade. São eles:

- AP 1.1 – O processo é executado;
- AP 2.1 – O processo é gerenciado;
- AP 2.2 – Os produtos de trabalho do processo são gerenciados;
- AP 3.1 – O processo é definido;
- AP 3.2 – O processo está implementado;
- AP 4.1 – O processo é medido;
- AP 4.2 – O processo é controlado;
- AP 5.1 – O processo é objeto de inovações;
- AP 5.2 – O processo é otimizado continuamente.

No nível E do MPS.BR se encontra o processo de avaliação e melhoria de processo organizacional. Esse processo é de fundamental importância para o sucesso dos demais processos, pois irá definir o quanto os processos contribuem para que a

empresa atinja os seus objetivos estratégicos de negócio. Esta etapa também irá avaliar a capacidade dos processos em apoiar o planejamento, realização e implantação das melhorias contínuas, que são identificadas com base nos pontos fracos e fortes dos processos (SOFTEX, 2009).

Com relação à avaliação de processos, o MPS.BR possui um Guia de Avaliação, que descreve todo o processo de avaliação, bem como o método de avaliação (MAMPS). Ambos foram baseados na norma ISO/IEC 15504-2:2003. O objetivo do método de avaliação é verificar a maturidade da organização ao executar os seus processos. Tanto o processo quanto o método foram definidos com os seguintes propósitos:

- permitir a avaliação objetiva dos processos de software de uma organização/unidade organizacional;
- permitir a atribuição de um nível de maturidade do MR-MPS com base no resultado da avaliação;
- ser aplicável a qualquer domínio de aplicação na indústria de software;
- ser aplicável a organizações/unidades organizacionais de qualquer tamanho.

Para atingir cada nível de maturidade, são avaliados todos os processos que compõem cada nível. Os processos são avaliados de acordo com os resultados esperados (REP) e também de acordo com os resultados de atributos de processo (RAP). Essa avaliação consiste em atribuir para cada um dos REPs e RAPs o seu grau de implementação, que pode ser:

T – Totalmente Implementado;
L – Largamente Implementado;
P – Parcialmente Implementado;
N – Não Implementado;
NA – Não Avaliado;
F – Fora de escopo.

Uma vez realizada a avaliação de todos os REPs e RAPs dos processos de um determinado nível, cabe à equipe de avaliação computar o grau de implementação dos processos da organização avaliada. A partir da definição desse grau de implementação, o resultado final da avaliação é apresentado pela equipe de avaliação para o representante da organização avaliada, indicando se a organização alcançou ou não o nível MPS desejado.

PARA COMPLEMENTAÇÃO DE ESTUDO

Acesse os sites oficiais da ISO[14], CMMI[15] e MPS.BR[16] e consulte as referências Chrissis (2003) e Rocha (2004) para complementar seu estudo sobre os modelos para melhoria dos processos de software.

REFERÊNCIAS BIBLIOGRÁFICAS

BARBARA, Saulo. *Gestão por Processos*. Rio de Janeiro: Qualitymark, 2006.

BECK, Kent. FOWLER, Martin. *Planning Extreme Programming*. 1. ed. Reading, MA: AddisonWesley, 2000.

BOEHM, B. W. A spiral model of software development and enhancement. IEEE Computer. 1988.

BOOCH, G. *Oriented-objected Analysis and Design with Applications*. Redwood City, CA: Benjamin Cummings. 1994.

BOOCH, G. *et al*. *The Unified Modeling Language User Guide*. Reading, MA: Addison Wesley Longman, 1999.

CONSTANTINE, L. L.; YOURDON, E. *Structured Design*. Englewood Cliffs, NJ: Prentice Hall, 1979.

CHRISSIS, M. B.; KONRAD, M.; SHRUM, S. *CMMI: Guidelines for software integration and product improvement*. [S. L.]: Addison Wesley, 2003. 752 p.

FOWLER, Martin, *et al*. *UML Essencial*. Terceira edição. Porto Alegre: Bookman, 2005.

FUGGETA, A. *A Classification of CASE Technology*. IEEE Computer, 1993.

GANE, C.; SARSON, T. *Structured System Analysis*. Englewood Cliffs, NJ: Prentice Hall, 1979.

HUFF, C. C. *Elements of a Realistic CASE Tool Adoption Budget*. *Comm*. ACM, 1992.

HUMPHREY, Watts. *Managing the Software Process*. Reading, MA: Addison-Wesley, 2004.

ISO/IEC 12207:2008 – *Systems and software engineering: software life cycle processes*. Geneva, 2008. Disponível em: <http://www.iso.org/iso/iso_catalogue/catalogue_ics/ catalogue_detail_ics.htm?csnumber =43447>. Acesso em: março de 2010.

ISO/IEC 15504:2008. *Tecnologia da informação* – Avaliação de processo.

JACKSON, M. A. *System Development*. Londres: Prentice Hall, 1983.

KRUCHTEN, Philippe. *Introdução ao RUP – Rational Unified Process*. Rio de Janeiro: Ciência Moderna, 2003, 255 p.

LARMAN, Craig. *Utilizando UML e Padrões. Uma introdução à análise e ao projeto orientados a objetos e ao desenvolvimento iterativo*. Porto Alegre: Bookman, 2007.

MARANHÃO, Mauriti. *Processo Nosso de Cada Dia, Modelagem de Processos de Trabalho*. Rio de Janeiro: Qualitymark, 2009.

NBR ISO/IEC 15504. *Tecnologia da informação – Avaliação de processo*. 2008.

OMG. *UML 2.0 Infraestructure Specification*. Object Management Group. 2003. http://www.uml.org.

OMG. *Update BPMN 2.0 Specification*. Object Management Group. 2010. http://www.bpmn.org.

PMI, PROJECT MANAGEMENT INSTITUTE. *A guide to the Project management body of knowledge – PMBOK 3rd edition*. Philadelphia: Project Management Institute, 2004.

PMI, PROJECT MANAGEMENT INSTITUTE. *Um Guia do conhecimento em gerenciamento de projetos – PMBOK*. 4. ed. Philadelphia: Project Management Institute, 2008.

PRESSMAN, Roger S. *Software Engineering*: a practitioner's approach. 5. ed. Boston: McGraw-Hill, 2001, 860 p.

ROBINSON, P. J. *Hierarchical Object-Oriented Design*. Englewood Cliffs, NJ: Prentice Hall, 1992.

ROCHA, A. R. C.; MALDONADO, J. C.; WEBER, K. C (orgs.). *Qualidade de software: teoria e prática*. 2. ed. São Paulo: Prentice Hall, 2004.

RUMBAUGH, J. *et al*. *Object-Oriented Modeling and Design*. Englewood Cliffs, NJ: Prentice Hall, 1991.

SOFTEX. *MPS.BR – Melhoria de Processo do Software Brasileiro*. 2009. Disponível em: <http://www.softex.br/mpsbr/guias/guias/ MPS.BRGuiaGeral2009.pdf>. Acesso em: outubro de 2010.

SOMMERVILLE, Ian. *Engenharia de Software*. 6. ed. São Paulo: Addison-Wesley, 2005, 592 p.

TELES, Vinícius Manhães. *Extreme Programming*. Rio de Janeiro: Novatec, 2004, 316 p. WHITTGN, Jeffrey L.; BENTLGY, Lonnie D. *System Analysis and Design Methods*. Richard D. Iwin, 1999.

1 ***Balanced Scorecard*** é uma metodologia de medição e gestão de desempenho criada em 1992. Os passos dessa metodologia incluem: definição da estratégia empresarial, gerência do negócio, gerência de serviços e gestão da qualidade; passos estes implementados através de indicadores de desempenho.

2 ***Benchmarking*** é um processo por meio do qual uma empresa examina como outra realiza uma função específica, a fim de melhorar como realizar a mesma ou uma função semelhante.

3 **ERP (*Enterprise Resource Planning*)** são sistemas integrados de gestão empresarial, ou seja, são sistemas de informação que integram todos os dados e processos de uma organização em um único sistema.

4 **ISO** (do inglês ***International Organization for Standardization***) é uma entidade que define a padronização e normalização para diversas áreas, em mais de 170 países.

5 **CMMI** (***Capability Maturity Model Integration***) é um modelo de referência para o processo de melhoria corporativo, baseado nas melhores práticas para desenvolvimento e manutenção de produtos, com ênfase em engenharia de software.

6 **MPS.BR** ou **Melhoria de Processos do Software Brasileiro** é simultaneamente um movimento para a melhoria da qualidade e um modelo de qualidade de processo voltado para a realidade do mercado de pequenas e médias empresas de desenvolvimento de software no Brasil.

7 http://www.bpminstitute.org

8 http://www.uml.org/

9 http://www.bpmn.org/

10 A orientação a objetos é um paradigma de análise, projeto e programação de sistemas de software baseado na composição e interação entre diversas unidades de software chamadas de objetos (LARMAN, 2007).

11 http://astah.change-vision.com/en/product/astah-community.html

12 http://www.bpmi.org

13 http://www.bizagi.com/

14 http://www.iso.org.br

15 http://www.sei.cmu.edu

16 http://www.softex.br/mpsbr

Editora da Universidade do Vale do Rio dos Sinos
EDITORA UNISINOS
Av. Unisinos,
950 93022-000 São Leopoldo RS Brasil

Telef: 51.3590 8239
Fax: 51.3590 8238
editora@unisinos.br

Souza, Vinícius Costa de.
Definindo o processo de software / Vinícius Costa de Souza. – São Leopoldo: Unisinos, 2011.

S729d

84 p.: il. – (Coleção EaDUnisinos).

Inclui bibliografia.
ISBN 978-85-7431-430-3

1. Padrões de software. I. Título. II. Série.

CDU 004.057.2

Dados Internacionais de Catalogação na Publicação (CIP)
(Bibliotecária Fabiane Pacheco Martino - CRB 10/1256)

Esta obra segue as normas do Acordo Ortográfico da Língua Portuguesa vigente desde 2009.

COLEÇÃO EAD

Editor
Carlos Alberto Gianotti

Acompanhamento editorial
Mateus Colombo Mendes

Revisão
Renato Deitos

Editoração
Rafael Tarcísio Forneck

Capa
Isabel Carballo

Impressão, verão de 2011

Sobre o autor

VINÍCIUS COSTA DE SOUZA é mestre em Computação Aplicada pela Universidade do Vale do Rio dos Sinos – UNISINOS (2005) e bacharel em Informática – habilitação em Análise de Sistemas pela UNISINOS (2002). Atua no ensino superior desde 2004 e possui experiência de dez anos no desenvolvimento de sistemas, principalmente baseados na Web. Ministra disciplinas da área de

Engenharia de Software nos cursos de graduação em Ciência da Computação e Sistemas de Informação e atua como coordenador executivo e professor da Graduação Tecnológica em Análise e Desenvolvimento de Sistemas e da Especialização em Qualidade de Software da UNISINOS.

Edição digital: dezembro 2013

Arquivo ePub produzido pela **Simplíssimo Livros**