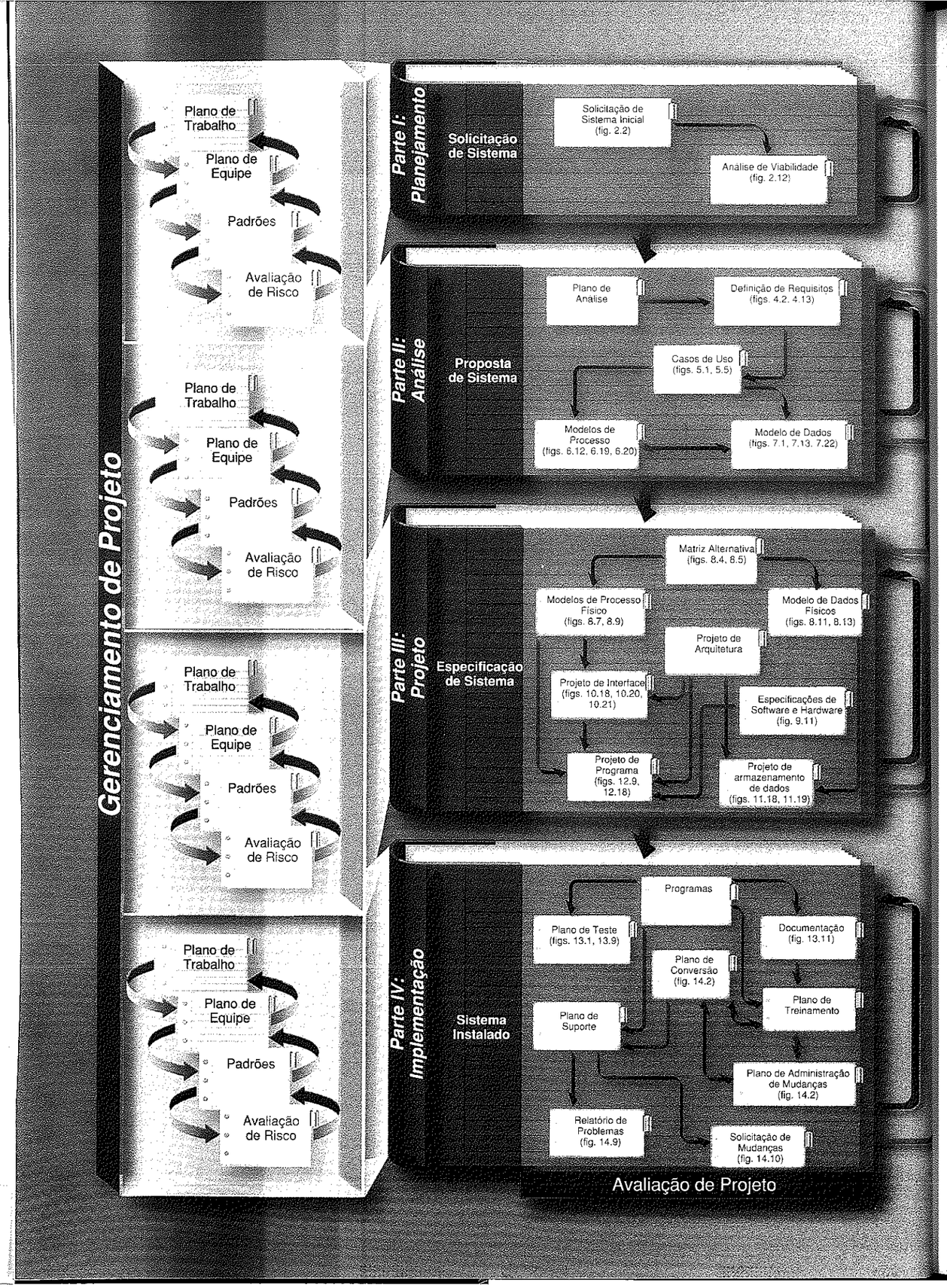


CAPÍTULO 1

INTRODUÇÃO

Este capítulo apresenta o ciclo de vida do desenvolvimento de sistemas, o modelo básico de quatro fases (planejamento, análise, projeto e implementação) que é comum a todos os projetos de desenvolvimento de sistemas. Ele examina, então, várias metodologias comumente usadas que diferem, em foco e abordagem, de cada uma daquelas fases. O capítulo se encerra com uma discussão das habilidades e papéis dentro da equipe de projeto.

OBJETIVOS

- Entender o conceito básico do ciclo de vida do desenvolvimento de sistemas e suas quatro fases.
- Entender os diversos e diferentes tipos de metodologias e como escolher entre eles.
- Familiarizar-se com as diferentes habilidades e papéis na equipe de projeto.

ESTRUTURA DO CAPÍTULO

Introdução
O Ciclo de Vida de Desenvolvimento de Sistemas
Planejamento
Análise
Projeto (Design)
Implementação
Metodologias de Desenvolvimento de Sistemas
Projeto Estruturado
Desenvolvimento de Aplicação Rápida
Desenvolvimento Ágil
Selecionando a Metodologia de Desenvolvimento Apropriada
Habilidades e Papéis da Equipe de Projeto
Analista de Negócios
Analista de Sistemas
Analista de Infra-estrutura
Analista de Gerenciamento de Mudança
Gerente de Projeto
Resumo

INTRODUÇÃO

O *ciclo de vida do desenvolvimento de sistemas* (SDLC, *systems development life cycle*) é o processo de compreensão de um sistema de informações (SI, *Information System*) que pode suprir as necessidades da empresa, projetar o sistema, construí-lo e entregá-lo aos usuários. Se você já teve aulas de programação ou já desenvolveu um programa por sua conta, isso provavelmente parecerá muito simples. Infelizmente, não é. Uma pesquisa realizada em 1996 pelo Standish Group descobriu que 42% de todos os projetos corporativos de SI foram abandonados antes da conclusão. Um estudo similar feito no mesmo ano pelo General Accounting Office constatou que 53% de todos os projetos de SI do governo dos Estados Unidos deixaram de ser concluídos. Infelizmente, grande parte dos sistemas que não são abandonados são entregues aos usuários com muito atraso, elevando seu custo acima do esperado e oferecendo menos recursos do que aqueles originariamente planejados.

Muitos de nós gostaríamos de achar que esses problemas só ocorrem com outras pessoas e empresas, mas eles acontecem com quase todas as empresas. Até a Microsoft tem um histórico de falhas e projetos com prazos vencidos (p. ex., Windows 1.0, Windows 95).[1]

Gostaríamos de promover este livro como um escudo que o manterá a salvo das falhas de SI, mas um escudo que garanta o sucesso do desenvolvimento de SI, não existe. Assim, este livro fornecerá vários conceitos básicos e muitas técnicas práticas que poderão ser usadas para melhorar a probabilidade de sucesso.

A pessoa-chave no SDLC é o analista de sistemas, que analisa a situação da empresa, identifica as oportunidades de melhorias e projeta um SI para implementá-las. O trabalho de um analista de sistemas é dos mais interessantes, excitantes e desafiadores. Como um analista de sistemas, você trabalhará com uma variedade de pessoas e aprenderá como elas conduzem a empresa. Você trabalhará com uma equipe de analistas de sistemas, programadores e outros profissionais em uma missão comum. Você sentirá a satisfação de ver que os sistemas que projetou e desenvolveu causaram um impacto significativo na empresa e saberá que contribuiu com sua capacidade para que aquilo acontecesse.

É importante lembrar que o objetivo principal do analista de sistemas não é criar um sistema maravilhoso. A finalidade principal é agregar valor para a organização, o que para a maioria das empresas significa aumentar os lucros (as agências do governo e organizações sem fins lucrativos medem seus resultados de maneira diferente). Muitos sistemas falharam e foram abandonados porque os analistas tentaram criar um sistema maravilhoso sem entender claramente como ele se adequaria aos objetivos da empresa, aos processos de negócios atuais e aos outros sistemas de informações para agregar valor. Um investimento em um sistema de informações é como qualquer outro investimento, como uma nova máquina-ferramental. O objetivo não é adquirir a ferramenta porque ela é simplesmente um meio para atingir um fim; o objetivo é permitir que a empresa execute melhor o seu trabalho, para que possa obter maiores lucros ou servir sua clientela de modo mais eficiente.

Este livro apresentará as habilidades básicas necessárias para que você se torne um analista de sistemas. Este é um livro pragmático que discute as melhores práticas no desenvolvimento de sistemas; ele não apresenta um estudo geral sobre desenvolvimento de sistemas que mostre a você tudo o que existe sobre o assunto. Por definição, os analistas de sistemas *executam procedimentos* e desafiam a maneira rotineira como as empresas trabalham. Para obter o máximo proveito deste livro, você precisará aplicar ativamente as idéias e conceitos nos exemplos constantes nos exercícios "Sua Vez", que são apresentados por todo o livro e, de maneira ideal, os exemplos desenvolvidos por você nos seus próprios projetos de desenvolvimento de sistemas. Este livro o orientará por todas as etapas do desenvolvimento de um sistema de informações bem-sucedido. Além disso, ilustraremos como uma organização (a qual chamaremos de CD Selections) aplica as etapas em um projeto (desenvolvendo um sistema de vendas de CD baseado na Web). Quando você terminar o livro, é lógico que não terá se tornado um analista experiente, mas estará pronto para começar a criar sistemas reais.

[1]Para obter mais informações sobre o problema, consulte Capers Jones, *Patterns of Software System Failure and Success*, London: Internationtal Thompson Computer Press, 1996; Capers Jones, *Assessment and Control of Software Risks*, Englewood Cliffs, NJ: Yourdon Press, 1994; Julia King, "IS Reins in Runaway Projects", *Computerworld*, February 24, 1997.

CONCEITOS EM AÇÃO

1-A Um Começo Falso e Dispendioso

Um grupo imobiliário no governo federal co-patrocionou um *data warehouse* com o departamento de tecnologia da informação (TI). Uma proposta formal foi escrita pelo TI, na qual os custos eram estimados em $800 mil, a duração do projeto foi estimada em oito meses e a responsabilidade dos recursos financeiros foi definida como sendo da unidade de negócios. O departamento de TI prosseguiu com o projeto antes de saber se a proposta já tinha sido aceita.

O projeto, na verdade, durou dois anos porque só a coleta dos dados demorou nove meses, em vez de um mês e meio, a base de usuários planejada cresceu de 200 para 2500 e o processo de aprovação para a compra da tecnologia para o projeto demorou um ano. Três semanas antes da entrega técnica, o diretor do TI cancelou o projeto. Esse esforço desperdiçado custou à empresa $2,5 milhões.

Fonte: "Data Warehousing Failure: Case Studies and Findings" *The Journal of Data Warehousing*, por Hugh J. Watson *et al.*, 4 (1), 1999, p.44-45.

Pergunta:

Por que esse sistema falhou? Por que a empresa gastou tempo e dinheiro em um projeto e, em seguida, o cancelou? O que poderia ter sido feito para evitar isso?

Neste capítulo, primeiro apresentaremos o SDLC básico seguido pelos projetos de SI. Esse ciclo de vida é comum para todos os projetos, embora o foco e a abordagem para cada fase do ciclo possam ser diferentes. Na próxima seção, discutiremos três tipos fundamentalmente diferentes de metodologias (projeto estruturado, desenvolvimento de aplicação rápida e desenvolvimento ágil). Finalmente, discutiremos um dos aspectos mais desafiadores do desenvolvimento de sistemas — a profundidade e a extensão das habilidades que são requeridas. Hoje, a maioria das empresas usa equipes de projeto cujos membros têm habilidades únicas, mas complementares. Este capítulo termina com uma discussão sobre os papéis-chave desempenhados pelos membros da equipe de desenvolvimento de sistemas.

O CICLO DE VIDA DE DESENVOLVIMENTO DE SISTEMAS

De muitas maneiras, a criação de um sistema de informações é similar à construção de uma casa. Em primeiro lugar, a casa (ou o sistema de informações) começa com uma idéia básica. Em segundo lugar, essa idéia é transformada em um desenho simples, que é mostrado ao cliente e refinado (freqüentemente por meio de vários desenhos, cada um deles aprimorando o outro), até que o cliente concorde que o desenho descreve exatamente o que ele quer. Em terceiro lugar, desenha-se um conjunto de plantas de instalações que apresenta as informações detalhadas sobre a casa (p. ex. planta de instalações hidráulicas, elétricas etc.). Finalmente, a casa é construída de acordo com essas plantas — e freqüentemente com algumas alterações e decisões tomadas pelo cliente enquanto a casa está sendo erguida.

O SDLC tem um conjunto similar de quatro *fases* fundamentais: planejamento, análise, projeto e implementação (Figura 1-1). Diferentes projetos podem enfatizar diferentes partes do SDLC ou abordar as fases do SDLC de diferentes maneiras. Cada fase é composta de uma série de *etapas*, que contam com *técnicas* que produzem *resultados* (*deliverables*) (arquivos de documento específicos que proporcionam a compreensão sobre o projeto).

Por exemplo, nos EUA, quando um estudante se candidata para uma universidade, existem várias fases que todos os alunos precisam percorrer: coleta de informações, inscrição e admissão. Cada uma dessas fases tem etapas: a coleta de informações inclui pesquisar as escolas, solicitar informações e ler os folhetos. Os alunos, então, usam recursos (p. ex., pesquisa na Internet) que podem ajudar na definição (p. ex., solicitar informações) de suas inscrições (p. ex. avaliações das universidades por diferentes aspectos).

A Figura 1-1 sugere que as fases e as etapas do SDLC prosseguem em um caminho lógico do início ao fim. Em alguns projetos isso é verdadeiro, mas em outros as equipes de projeto se movem por meio de etapas consecutivas e iterativamente, ou em outros padrões. Nesta seção, descreveremos as fases e as etapas além de algumas técnicas que são usadas para executar essas etapas em um nível muito alto. Devemos enfatizar que nem todas as empresas seguem o SDLC exatamente como descreveremos a seguir. Como veremos em breve, existem muitas variações sobre o SDLC global.

Fase	Capítulo	Etapa	Técnica	Resultado
	2	Identificar Oportunidade	Identificação do Projeto	Solicitação de Sistema
Planejamento (Por que construir o sistema?)	2	Analisar Viabilidade	Viabilidade Técnica Viabilidade Econômica Viabilidade Organizacional	Análise de Viabilidade
Solicitação de Sistema	3	Desenvolver Plano de Trabalho	Estimativa de Tempo *Timeboxing* Identificação de Tarefas Estrutura de Divisão de Trabalho Gráfico Pert Gráfico GANTT Gerenciamento de Escopo	Plano de Trabalho
	3	Criar Equipe de Projeto	Equipe de Projeto Organograma do Projeto	Plano de Equipe
	3	Controlar e Dirigir Projeto	Repositório CASE Padrões Documentação Gerenciamento de Risco	Lista de Padrões Avaliação de Risco
Análise (Quem, o quê, quando, onde será o sistema)	4	Desenvolver Estratégia de Análise	Automação do Processo de Negócios Melhoramento do Processo de Negócios Reengenharia do Processo de Negócios	Proposta de Sistema
Proposta de Sistema	4	Determinar Requisitos da Empresa	Entrevista Sessão JAD Questionário Análise de Documentos Observação	Definição de Requisitos
	5	Criar Casos de Uso	Análise de Caso de Uso	Casos de Uso
	6	Modelar Processo	Diagramação de Fluxo de Dados	Modelos de Processo
	7	Modelar Dados	Modelagem de Relacionamento de Entidade Normalização	Modelo de Dados
	8	Projetar Sistema Físico	Seleção de Projeto Diagramação de Fluxo de Dados Modelagem de Relacionamento de Entidade	Matriz de Alternativa Modelos de Processo Físico Modelo de Dados Físicos
Projeto (Como o sistema funcionará?)	9	Projetar Arquitetura	Projeto de Arquitetura Seleção de Hardware e Software	Especificação de Sistema Relatório de Arquitetura Especificação de Hardware e Software
Especificação de sistema	10	Projetar Interface	Cenário de Uso Estrutura de Interface Padrões de Interface Protótipo de Interface Avaliação de Interface	Projeto de Interface
	11	Projetar Bancos de Dados e Arquivos	Seleção de Formato de Dados Desnormalização Ajuste de Desempenho Estimativa de Tamanho	Projeto de Armazenamento de Dados
	12	Projetar Programas	Análise de Transformação Gráfico de Estrutura de Programa Especificação de Programa	Projeto de Programa
Implementação (Entrega do sistema)	13	Construir Sistema	Programação Teste de Software Teste de Desempenho	Plano de Teste Programas Documentação
Sistema instalado	14	Instalar Sistema	Seleção de Estilo de Conversão Treinamento	Plano de Conversão Plano de Treinamento
	14	Manter Sistema	Seleção de Suporte Manutenção do Sistema Avaliação de Projeto	Plano de Suporte Relatório de Problemas Solicitação de Mudança
	14	Pós-implementação	Auditoria de Pós-implementação	Relatório de Auditoria de Pós-implementação

FIGURA 1-1
Fases do Ciclo de Desenvolvimento dos Sistemas

Por enquanto, existem dois pontos importantes sobre o SDLC que precisam ser entendidos. Primeiro, você deve obter um sentido geral das fases e etapas que os projetos de SI percorrem e algumas das técnicas que produzem determinados resultados. Segundo, é importante entender que o SDLC é um processo de *refinamento gradual*. Os resultados produzidos na fase de análise proporcionam uma idéia geral da forma do novo sistema. Esses resultados são usados como entrada para a fase de projeto, a qual os refina para produzir um conjunto de resultados que descrevem em termos muito mais detalhados o modo como o sistema será construído. Esses resultados, por sua vez, são usados na fase de implementação para produzir o sistema real. Cada fase refina e elabora o trabalho feito anteriormente.

Planejamento

A *fase de planejamento* é o processo fundamental para compreender *por que* um sistema de informações deve ser construído e determinar como a equipe de projeto trabalhará para construí-lo. A primeira etapa é identificar as oportunidades, quando o valor agregado do sistema para a empresa será conhecido — como ele diminuirá os custos e aumentará os lucros? Grande parte das idéias para novos sistemas vem de fora da área de SI (do departamento de marketing, do departamento de contabilidade etc.) na forma de uma *solicitação de sistema*. Uma solicitação de sistema apresenta um breve resumo de uma necessidade da empresa e explica como um sistema que suporta a necessidade agregará valor. O departamento de SI trabalha com a pessoa ou departamento que gerou a solicitação (denominado *responsável pelo projeto*) para conduzir uma *análise de viabilidade* que examina a viabilidade técnica da idéia (isto é, podemos construí-lo?), a viabilidade econômica (ele agregará valor?) e a viabilidade organizacional (se o construirmos, ele será usado?).

A solicitação de sistema e a análise de viabilidade são apresentadas a um *comitê de aprovação* de SI (às vezes chamado de comitê de sugestões), o qual decide se o projeto deve ser empreendido. Se o comitê aprova o projeto, então a próxima etapa do planejamento ocorre — o *gerenciamento do projeto*. Durante o gerenciamento do projeto, o *gerente do projeto* cria um *plano de trabalho*, aloca o pessoal para o projeto e coloca técnicas no lugar certo para ajudar a controlá-lo e direcioná-lo durante todo o ciclo de vida de desenvolvimento do projeto. O produto do gerenciamento de projeto é um *plano de trabalho do projeto* que descreve como a equipe de projeto executará o desenvolvimento do sistema. Esse plano de trabalho inclui todos os resultados do gerenciamento de projeto.

Análise

A *fase de análise* responde a perguntas sobre *quem* usará o sistema, *o que* o sistema fará e *onde* e *quando* ele será usado. Durante essa fase, a equipe de projeto investiga todos os sistemas atuais, identifica as oportunidades de aperfeiçoamento e desenvolve um conceito para o novo sistema. Veja a Figura 1-1.

A análise começa com o desenvolvimento de uma *estratégia de análise* que orienta os esforços da equipe de projeto. Tal estratégia normalmente inclui uma análise do sistema atual (denominado *sistema no estado*) e seus problemas e, em seguida, uma análise das maneiras de projetar um novo sistema (denominado *sistema futuro*). A próxima etapa é reunir os requisitos do negócio (p. ex., por meio de entrevistas ou questionários). A análise dessas informações — junto com a entrada do responsável do projeto e muitas outras pessoas — leva ao desenvolvimento de uma lista de requisitos de negócios para o novo sistema. Essa Definição de Requisitos é, então, usada como uma base para desenvolver casos de uso e um *modelo de processos* que descreve o modo como a empresa operará, se o novo sistema for desenvolvido. A Figura 1-2 mostra um exemplo do modelo de processos, com processos (as tarefas que alguém executa) mostrados como caixas de cantos arredondados, as entradas e saídas dos processos como setas e os contêineres de armazenamento de dados como caixas com um lado aberto. Finalmente, um *modelo de dados* é desenvolvido para descrever as informações que são necessárias para apoiar o processo.

A análise, a Definição de Requisitos para os Casos de Uso, o modelo de processos e o modelo de dados são combinados em um documento denominado *proposta de sistema*, que é apresentado ao responsável pelo projeto e a outros importantes tomadores de decisão (p. ex., membros do comitê de aprovação), que decidem se o projeto deve seguir adiante. Veja a Figura 1-1.

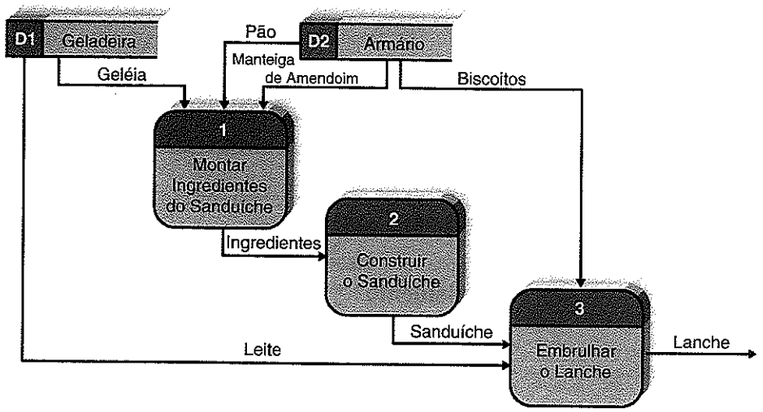


FIGURA 1-2
Um Modelo de Processo Simples para Fazer um Lanche

A proposta do sistema é o resultado inicial que descreve quais requisitos da empresa que o novo sistema deve atender. Como essa é, na verdade, a primeira etapa do projeto do novo sistema, alguns especialistas argumentam que não é apropriado usar o termo *análise*, como o nome para essa fase, e argumentam que o melhor nome seria *análise e projeto inicial*. Como a maioria das empresas continua a usar o nome *análise* para essa fase, nós o usaremos também neste livro, mas é importante lembrar que o resultado da fase de análise é uma análise e um projeto inicial de alto nível para o novo sistema.

Projeto

A *fase de projeto* decide *como* o sistema operará, em termos de infra-estrutura de hardware, software e rede; a interface do usuário, os formulários e os relatórios que serão usados; e os programas, bancos de dados e arquivos específicos que serão necessários. Embora a maior parte das decisões estratégicas sobre o sistema tenha sido tomada no desenvolvimento da concepção do sistema durante a fase de análise, as etapas na fase de projeto determinam exatamente como o sistema operará.

A primeira etapa na fase de projeto é executar a *seleção de projeto*: o sistema será desenvolvido pelos próprios programadores da empresa, será terceirizado para outra firma (normalmente uma firma de consultoria) ou a empresa comprará um pacote de software existente. Isso leva ao desenvolvimento do *projeto de arquitetura* básico do sistema que descreve a infra-estrutura de hardware, software e rede que será usada. Na maioria dos casos, o sistema adicionará ou alterará a infra-estrutura já existente na organização. O *projeto de interface* especifica como os usuários se moverão pelo sistema (isto é, métodos de navegação, como menus e botões na tela) e os formulários e relatórios que o sistema usará. Em seguida, é desenvolvido o *projeto de armazenamento* de dados, que define exatamente quais e como os dados serão armazenados. Finalmente, a equipe de análise desenvolve o projeto de programa, que define os programas que precisam ser escritos e exatamente o que cada programa fará.

Essa coleção de resultados (projeto de arquitetura, projeto de interface, especificações de banco de dados e de arquivo e projeto de programa) é a *especificação do sistema* que será fornecida à equipe de programação para implementação. No final da fase de projeto, a análise de viabilidade e o plano de trabalho são reexaminados e revisados, e outra decisão sobre a continuação ou não do projeto é tomada pela pessoa responsável pelo projeto e pelo comitê de aprovação. Veja a Figura 1-1.

Implementação

A fase final do SDLC é a *fase de implementação*, durante a qual o sistema é realmente construído (ou comprado, no caso de um projeto de software de terceiros). Essa é a fase que normalmente exige mais atenção porque, para a maioria dos sistemas, é a maior e mais cara parte do processo de desenvolvimento.

A primeira etapa da implementação é a *construção* do sistema, durante a qual o sistema é construído e testado para garantir que funcione de acordo com o projeto. A fase de testes é uma das etapas mais críticas na implementação, porque os custos dos erros (*bugs*) podem ser imensos; grande parte das empresas gasta mais tempo e atenção na fase de testes do que escrevendo os programas. Uma vez que o sistema tenha passado por uma série de testes, ele é instalado. A *instalação* é o processo pelo qual o sistema antigo é desativado e o novo é ativado, e ela pode incluir um enfoque de substituição direta (na qual o novo sistema substitui imediatamente o sistema antigo), uma abordagem paralela (na qual os sistemas novo e antigo são operados por um mês ou dois até que se tenha certeza de que não há nenhum erro no novo sistema) ou uma estratégia de conversão por fases (na qual o novo sistema é instalado em uma parte da empresa como uma experiência inicial e, em seguida, gradualmente instalado em outras). Um dos aspectos mais importantes da conversão é o desenvolvimento de um *plano de treinamento* para ensinar aos usuários como usar o novo sistema e ajudar a gerenciar as alterações causadas por ele.

Uma vez que o sistema tenha sido instalado, a equipe de analistas estabelece um *plano de suporte* para o sistema. Este plano inclui normalmente uma revisão formal e informal de pós-implementação, bem como um modo sistemático para identificar as necessidades de alteração principais e secundárias do sistema.

METODOLOGIAS DE DESENVOLVIMENTO DE SISTEMAS

Uma *metodologia* é um enfoque formalizado para implementar o SDLC (isto é, uma lista de etapas e resultados). Existem muitas metodologias de desenvolvimento de sistemas diferentes, e cada uma delas é única devido à sua ênfase no processo *versus* dados, além da ordem e do foco que ela coloca em cada fase do SDLC. Algumas metodologias são padrões formais usados por agências do governo, embora outras tenham sido desenvolvidas por firmas de consultoria para serem vendidas aos clientes. Muitas empresas têm suas próprias metodologias internas que foram refinadas durante anos e explicam exatamente como cada fase do ciclo de vida do desenvolvimento de sistema deve ser executada naquela empresa.

Existem muitas maneiras de classificar as metodologias. Uma maneira é examinar se ela enfoca os processos de negócios ou os dados que dão suporte aos negócios. As metodologias são *centradas no processo* se enfatizam modelos de processos (Capítulo 6) como o ponto principal da concepção do sistema. Na Figura 1-2, por exemplo, as metodologias centradas em processo focalizarão primeiro a definição do processo (p. ex., a montagem dos ingredientes de um sanduíche). As metodologias são *centradas em dados* se enfatizam os modelos de dados (Capítulo 7) como o ponto principal da concepção do sistema. Na Figura 1-2, por exemplo, as metodologias centradas em dados poderiam focalizar a definição do conteúdo das áreas de armazenamento (p. ex., o refrigerador) e como os conteúdos foram organizados. Outras metodologias, como as *orientadas a objeto* (Capítulo 15), tentam equilibrar o foco entre o processo e os dados incorporando ambos em um único modelo.[2] Na Figura 1-2, essas metodologias poderão focalizar primeiro a definição dos elementos principais do sistema (isto é, sanduíches, almoços) e examinar os processos e os dados (os itens armazenados) que estavam envolvidos em cada um.

Outro fator importante na classificação das metodologias é a seqüência de fases do SDLC e a quantidade de tempo e esforço devotados a cada uma delas.[3] Quando a computação estava começando, a necessidade de metodologias de ciclo de vida formais e bem planejadas não era bem entendida. Os programadores tinham a tendência de se mover diretamente de uma fase de planejamento muito simples para a etapa de construção da fase de implementação; em outras palavras, eles iam diretamente da solicitação do sistema, ainda confusa e sem planejamento, para a codificação, que é o mesmo enfoque que você às vezes usa quando escreve programas para uma classe

[2]A metodologia moderna e clássica centrada em processo é a de Edward Yourdon, *Modern Structured Analysis*, Englewood Cliffs, NJ: Yourdon Press, 1989. Um exemplo de uma metodologia centrada em dados é a engenharia de informações de James Martin, *Information Engineering*, volumes 1-3, Englewood Cliffs, NJ: Prentice Hall, 1989. Muitas novas metodologias orientadas a objetos são baseadas na Unified Modeling Language definida no *UML Document Set*, Santa Clara, CA: Relational Software Corp., 1997. Uma metodologia padronizada amplamente aceita que equilibra processos e dados é IDEF; consulte FIPS 183, *Integration Definition for Function Modeling*. Federal Information Processing Standards Publications, Washington, D.C.: U.S. Department of Commerce, 1993.

[3]Uma boa referência para comparar as metodologias de desenvolvimento de sistemas é Steve McConnell, *Rapid Development*, Redmond, WA: Microsoft Press, 1996.

de programação. Esse enfoque pode funcionar para pequenos programas que requerem apenas um programador, mas se as exigências são complexas ou pouco claras você poderá omitir aspectos importantes do problema e precisar começar tudo de novo, desperdiçando parte do programa (e o tempo e o esforço gastos para escrevê-lo). Esse enfoque também dificulta o trabalho em equipe, porque os membros têm pouca idéia do que precisa ser executado e da maneira como podem trabalhar juntos para conseguir um produto final.

Projeto Estruturado

A primeira categoria de metodologias de desenvolvimento de sistemas é denominada *projeto estruturado*. Essas metodologias se tornaram dominantes na década de 1980, substituindo o enfoque anterior geral e indisciplinado. As metodologias de projeto estruturado adotam um enfoque formal passo a passo para o SDLC que se move logicamente de uma fase para a outra. Existem várias metodologias centradas em processo e metodologias centradas em dados que seguem o enfoque básico das duas categorias de projeto estruturado descrito a seguir.

Desenvolvimento em Cascata A metodologia de projeto estruturado original (que é usada até hoje) é o *desenvolvimento em cascata* (*waterfall development*). Com o desenvolvimento em cascata, o analista e os usuários prosseguem em seqüência de uma fase para outra; Veja a Figura 1-3. Os resultados importantes para cada fase são em geral produzidos no papel (freqüentemente centenas de páginas) e são apresentados ao responsável pelo projeto para aprovação, à medida que o projeto se move de fase para fase. Uma vez que o responsável aprova o trabalho que foi conduzido para uma fase, esta termina e a próxima começa. Esse enfoque é denominado desenvolvimento em cascata porque se move para frente de fase para fase, da mesma maneira que uma cascata. Embora seja possível voltar no ciclo de vida do desenvolvimento de sistemas (p. ex., voltar do projeto para a análise), isso é extremamente difícil. (Imagine-se como um salmão tentando nadar para cima em uma cascata como mostra a Figura 1-3.)

As duas principais vantagens do desenvolvimento em cascata são que ele identifica os requisitos do sistema, antes de a programação começar, e minimiza as alterações feitas nos requisitos, à medida que o projeto prossegue. As duas desvantagens principais são que o projeto deve ser completamente especificado no papel, antes de a programação começar, e transcorre um longo tempo entre a conclusão da proposta do sistema na fase de análise e a entrega do sistema (normalmente, muitos meses ou anos). Um documento de papel é freqüentemente um mecanismo de comunicação pobre; portanto, os requisitos importantes podem não ser notados em centenas de páginas de documentação. Os usuários raramente são apresentados adequadamente ao novo sistema, o que ocorre muito depois que a idéia inicial do sistema foi introduzida. Se a equipe de projeto deixa passar requisitos importantes, talvez seja necessária uma programação de implementação posterior muito cara. (Imagine-se tentando projetar um carro no papel. Como você ficaria ao ter de se

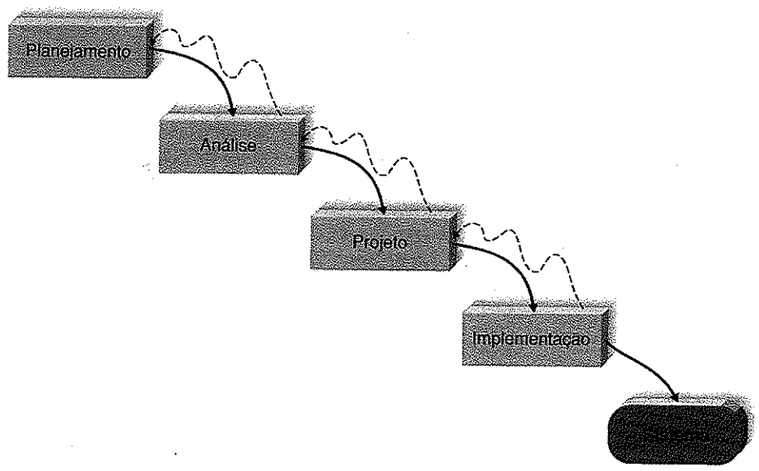


FIGURA 1-3
A Metodologia de Desenvolvimento em Cascata

lembrar de incluir as luzes internas que acendem quando as portas se abrem ou especificar o número certo de válvulas do motor?)

Um sistema também pode exigir uma significativa repetição de trabalho, porque o ambiente empresarial mudou desde a hora em que a fase de análise começou. Quando ocorrem alterações, isso significa voltar para as fases iniciais e seguir a alteração através de cada uma das fases subseqüentes sucessivamente.

Desenvolvimento Paralelo A metodologia do *desenvolvimento paralelo* tenta tratar o problema dos longos atrasos entre a fase de análise e a entrega do sistema. Em vez de fazer o projeto e a implementação em seqüência, ele executa um projeto geral para o sistema inteiro e, em seguida, o divide em uma série de subprojetos distintos que podem ser projetados e implementados em paralelo. Uma vez que todos os subprojetos estejam concluídos, há uma integração final das partes separadas, e o sistema é entregue (Figura 1-4).

A principal vantagem dessa metodologia é que ela pode reduzir o tempo de planejamento necessário para entregar um sistema; portanto, há menos chance de as alterações que ocorrem no ambiente da empresa causarem uma repetição de trabalho. Entretanto, o enfoque ainda sofre dos problemas provocados pelos documentos de papel. Ele também adiciona um novo problema: às vezes os subprojetos não são completamente independentes; as decisões de projeto tomadas em um subprojeto podem afetar outro, e o final do projeto talvez exija significativos esforços de integração.

Desenvolvimento de Aplicação Rápida

O *desenvolvimento de aplicação rápida* (*RAD*, *rapid application development*) é um enfoque recente para o desenvolvimento de sistemas surgido nos anos 1990. O RAD tenta lidar com duas fragilidades das metodologias de desenvolvimento estruturado: os períodos de desenvolvimento muito longos e a dificuldade de entender um sistema a partir de uma descrição baseada em papel. As metodologias RAD ajustam as fases do ciclo de vida de desenvolvimento do sistema para que

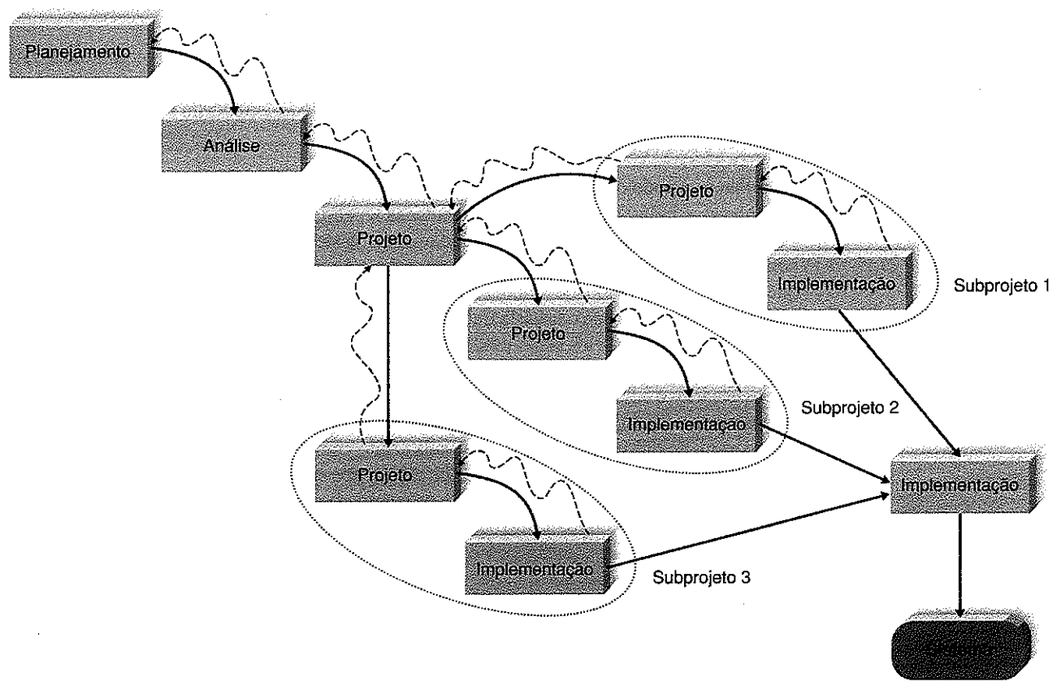


FIGURA 1-4
A Metodologia de Desenvolvimento Paralelo

alguma parte do sistema seja rapidamente desenvolvida e vá logo para as mãos dos usuários. Dessa maneira, os usuários podem entender melhor o sistema e sugerir revisões para que ele possa atender satisfatoriamente às suas necessidades.[4]

A maioria das metodologias RAD recomenda que os analistas usem técnicas especiais e ferramentas de computador para acelerar as fases de análise, projeto e implementação, como as ferramentas CASE (*computer-aided software engineering*) (veja o Capítulo 3), as sessões JAD (*joint application design*) (consulte o Capítulo 5), as linguagens de programação visual de quarta geração que simplificam e aceleram a programação (p. ex., o Visual Basic) e os geradores de código que automaticamente produzem programas a partir das especificações do projeto. É a combinação das fases alteradas do SDLC e do uso dessas ferramentas e técnicas que melhora a velocidade e a qualidade do desenvolvimento dos sistemas. Existem metodologias centradas em processos, centradas em dados e orientadas a objetos que seguem esses enfoques básicos das três categorias de RAD descritas a seguir.

Desenvolvimento em Fases A metodologia do *desenvolvimento em fases* divide o sistema global em uma série de *versões* que são desenvolvidas seqüencialmente. A fase de análise identifica o conceito de sistema global, e a equipe de projeto, os usuários e o responsável pelo sistema categorizam os requisitos em uma série de versões. Os requisitos mais importantes e fundamentais são reunidos na primeira versão do sistema. A fase de análise, então, leva ao projeto e à implementação, mas apenas com o conjunto de requisitos identificado para a versão 1 (Figura 1-5).

Uma vez que a versão 1 é implementada, começa o trabalho sobre a versão 2. A análise é executada com base nos requisitos identificados anteriormente e combinada com novas idéias e questões que surgem da experiência dos usuários com a versão 1. A versão 2, então, é projetada e implementada, e o trabalho sobre a próxima versão começa imediatamente. Esse processo continua até que o sistema esteja concluído ou que não esteja mais em uso.

O desenvolvimento em fases tem a vantagem de colocar rapidamente um sistema útil nas mãos dos usuários. Embora de imediato ele não execute todas as funções, ele começa a agregar valor mais rápido que se o sistema fosse entregue após a conclusão, como é o caso das metodologias em cascata ou paralela. Da mesma maneira, como os usuários começam a trabalhar com o sistema mais cedo, é mais provável que identifiquem requisitos adicionais importantes mais cedo do que em situações de projeto estruturado.

A principal desvantagem do desenvolvimento em fases é que os usuários começam a trabalhar com sistemas que estão intencionalmente incompletos. É crítico identificar os recursos mais importantes e úteis e incluí-los na primeira versão, enquanto se gerenciam as expectativas dos usuários ao longo do caminho.

Protótipo A metodologia de *protótipo* executa as fases de análise, projeto e implementação de modo simultâneo, e todas as três fases são executadas repetidamente em ciclo até que o sistema esteja concluído. Com essa abordagem, os fundamentos básicos de análise e projeto são executados e começa imediatamente o trabalho sobre um *protótipo do sistema*, um programa "rápido e sujo" que fornece uma quantidade mínima de recursos. O primeiro protótipo é normalmente a primeira parte do sistema com que o usuário trabalhará. Isso é mostrado aos usuários e ao responsável pelo projeto, cujos comentários serão usados para reanalisar, reprojetar e reimplementar um segundo protótipo que fornece alguns recursos a mais. Esse processo continua em ciclo até que os analistas, os usuários e o responsável concordem que o protótipo fornece funcionalidade suficiente para ser instalado e usado na empresa. Depois que o protótipo (agora chamado de "sistema") é instalado, ocorrem os refinamentos até que ele seja aceito como o novo sistema (Figura 1-6).

A vantagem principal do protótipo é que ele fornece *muito* rapidamente um sistema com o qual os usuários podem interagir, mesmo que a princípio ele não esteja pronto para ser usado no âmbito empresarial. O protótipo mostra aos usuários que a equipe de projeto está trabalhando no sistema (não há mais atrasos, vistos pelos usuários como pouco progresso), e a abordagem ajuda a refinar mais rapidamente os requisitos reais. Em vez de tentar entender uma especificação do sistema no papel, os usuários podem interagir com o protótipo para entender melhor o que ele pode ou não fazer.

[4]Um dos melhores livros sobre a metodologia RAD é aquele escrito por Steve McConnell, *Rapid Development*, Redmond WA: Microsoft Press, 1996.

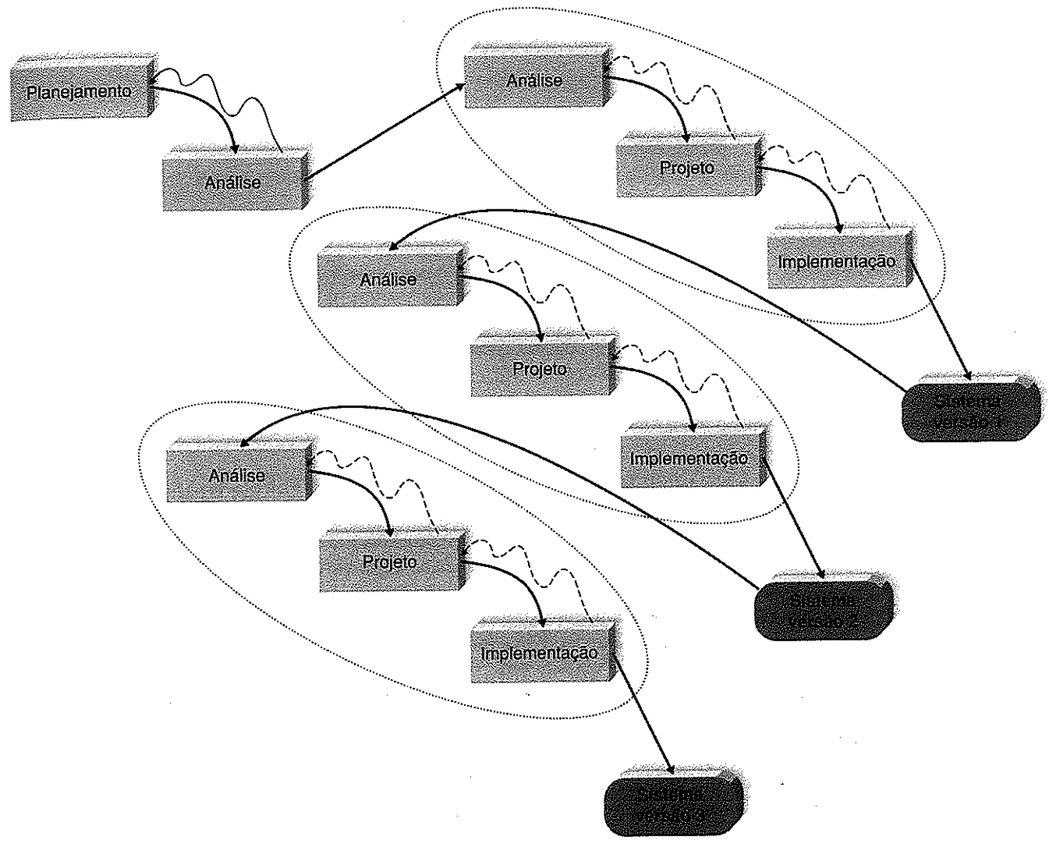


FIGURA 1-5
A Metodologia de Desenvolvimento em Fases

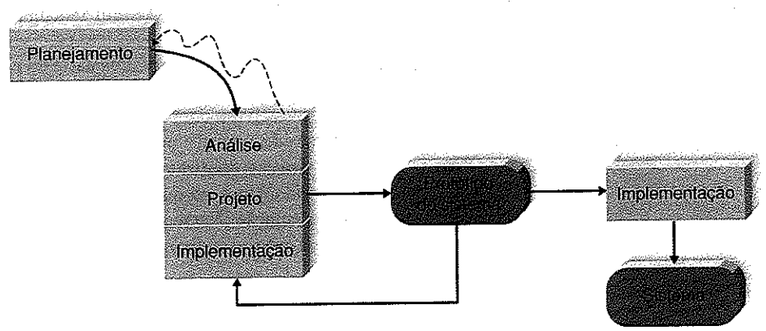
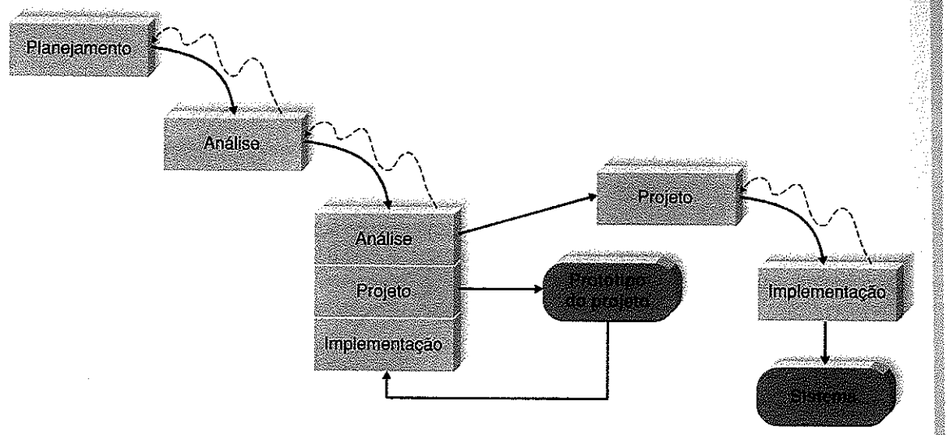


FIGURA 1-6
A Metodologia de Protótipo

FIGURA 1-7
A Metodologia de Protótipo Descartável



O problema principal do protótipo é que suas versões de sistema de passos rápidos desafiam as tentativas de conduzir uma análise cuidadosa e metódica. Freqüentemente o protótipo é submetido a tantas alterações significativas que muitas decisões iniciais do projeto se tornam decisões desfavoráveis. Isso pode causar problemas no desenvolvimento de sistemas complexos, porque as questões e os problemas principais não são reconhecidos até que o processo já esteja bem desenvolvido. Imagine construir um carro e descobrir no processo de protótipo que você tem de retirar todo o motor para trocar o óleo (porque ninguém pensou na necessidade de trocar o óleo até ele já ter rodado 16 mil quilômetros).

Protótipo Descartável A metodologia de *protótipo descartável* (*throwaway prototyping*) é similar à metodologia de protótipo no que diz respeito ao desenvolvimento de protótipos; entretanto, os protótipos descartáveis são feitos em um ponto diferente no SDLC. Esses protótipos são usados para um objetivo muito diferente daqueles discutidos anteriormente, e têm uma aparência também muito diferente[5] (Figura 1-7).

O protótipo descartável tem uma fase de análise meticulosa que é usada para reunir informações e desenvolver idéias para a concepção do sistema. Entretanto, muitos dos recursos sugeridos pelos usuários talvez não sejam bem entendidos, e poderá haver questões técnicas desafiantes a serem resolvidas. Cada uma dessas questões é examinada pela análise, pelo projeto e pela construção de um *protótipo de projeto*. Um protótipo de projeto não é um sistema de trabalho; ele apenas representa uma parte do sistema que precisa de refinamentos adicionais e contém somente os detalhes suficientes para permitir que os usuários entendam as questões sob consideração. Por exemplo, suponha que não tenha ficado completamente claro para os usuários como um sistema de entrada de pedidos deve trabalhar. A equipe de análise poderá criar uma série de páginas HTML visualizadas com um navegador Web para ajudar os usuários a visualizar esse sistema. Nesse caso, uma série de telas do protótipo *parecerá* ser um sistema, mas na verdade elas não fazem nada. Como alternativa, suponha que a equipe de projeto precise desenvolver um programa gráfico sofisticado em Java. A equipe poderá escrever uma parte do programa com os dados simulados para garantir que o programa seja bem-sucedido.

Um sistema que é desenvolvido com o uso desse enfoque normalmente usa vários protótipos de projeto durante as fases de análise e projeto. Cada um desses protótipos é usado para minimizar o risco associado ao sistema, confirmando que questões importantes serão entendidas antes de o sistema real ser construído. Uma vez que as questões estiverem resolvidas, o projeto caminhará para o design e a implementação. Nesse ponto, os protótipos de projeto são descartados, o que é uma diferença importante entre essa abordagem e o protótipo, no qual os protótipos evoluem para o sistema final.

[5]Nossa descrição da metodologia de protótipo descartável é uma versão modificada da metodologia de desenvolvimento em espiral desenvolvida por Barry Boehm, "A Spiral Model of Software Development and Enhancement", *Computer*, maio 1988, 21(5):61-72.

O protótipo descartável equilibra os benefícios das fases de análise e projeto bem fundamentadas com as vantagens do uso de protótipos para refinar questões importantes antes de um sistema ser construído. O sistema final poderá demorar mais para ser entregue se comparado com o protótipo (porque os protótipos não se transformam no sistema final), mas o enfoque normalmente produz sistemas mais estáveis e confiáveis.

Desenvolvimento Ágil

A terceira categoria de metodologias de desenvolvimento de sistemas ainda está surgindo, e é denominada *Desenvolvimento Ágil* (*Agile Development*). Essa metodologia centrada na programação tem poucas regras e práticas, todas fáceis de seguir. Elas enfocam o aperfeiçoamento do SDLC, eliminando grande parte do excesso de modelos e de documentação e o tempo gasto nessas tarefas. Em vez disso, os projetos enfatizam um desenvolvimento de aplicação simples e iterativo. Os exemplos das metodologias de Desenvolvimento Ágil incluem a Extreme Programming (programação extrema), scrum e o DSDM (*dynamic systems development method*, método de desenvolvimento de sistemas dinâmico). O enfoque do Desenvolvimento Ágil, conforme descrito a seguir, normalmente é usado junto com as metodologias orientadas a objeto.

Extreme Programming A *extreme programming (XP)*[6] usa codificação simples e teste contínuo executados por dois desenvolvedores, e interações estreitas com os usuários finais para construir sistemas muito rapidamente. Após um processo de planejamento superficial, os projetos executam as fases de análise, projeto e implementação iterativamente (Figura 1-8). A funcionalidade do sistema cresce ao longo do tempo.

As práticas de teste e codificação eficiente são o ponto-chave da XP. Na verdade, cada código é testado e colocado em um ambiente de teste integrante. Se houver *bugs*, o código é revisto até ficar completamente livre de erros. A XP conta pesadamente com o *refactoring*, que é um modo disciplinado de reestruturar o código para mantê-lo simples.

Um projeto XP começa com histórias de usuários que descrevem o que o sistema precisa fazer. Depois, os programadores codificam módulos e testes pequenos e simples para atender àquelas necessidades. Os usuários devem estar disponíveis para esclarecer questões e tópicos à medida que eles surgirem. Os padrões são muito importantes para minimizar a confusão, portanto as equipes XP usam um conjunto de nomes, descrições e práticas de codificação comuns.

Os projetos XP entregam os resultados mais cedo até que as abordagens RAD, e raramente têm problemas para reunir os requisitos do sistema. A XP trabalha muito bem com tecnologias orientadas a objetos. A metodologia trabalha bem com projetos que têm agendas muito curtas ou prazos críticos.

Entretanto, a XP requer muita disciplina. Caso contrário, os projetos podem se tornar sem foco e caóticos. Ela é recomendada para pequenos grupos de desenvolvimento — não mais que doze desenvolvedores trabalhando em dupla, e não é recomendada para aplicações grandes. Além disso, a metodologia precisa de muitas entradas de usuários por meio do site, algo com o qual muitas unidades da empresa não podem se comprometer.

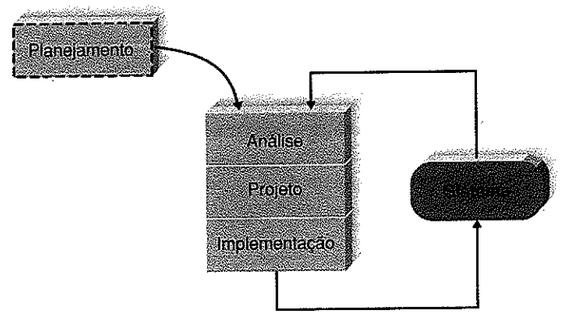


FIGURA 1-8
A Metodologia Extreme Programming

[6]Para obter mais informações, consulte *Extreme Programming Explained: Embrace Change*, City: Addison-Wesley, 1999.

Selecionando a Metodologia de Desenvolvimento Apropriada

Como existem muitas metodologias, o primeiro desafio que os analistas enfrentam é selecionar qual metodologia usar. A escolha da metodologia não é simples, porque nenhuma delas é sempre a melhor. (Se fosse, nós simplesmente a usaríamos em todos os projetos!) Muitas empresas têm padrões e políticas que orientam a escolha da metodologia. Você verá que as empresas podem ter uma ou várias opções de metodologia aprovadas sem ter nenhuma política formal.

A Figura 1-9 resume alguns critérios importantes de seleção de metodologia. Um item essencial, não discutido nessa figura, é o grau de experiência da equipe de análise. Grande parte das metodologias RAD requer o uso de novas ferramentas e tecnologias que têm uma curva de aprendizado significativa. Freqüentemente, essas ferramentas e técnicas aumentam a complexidade do projeto e exigem um tempo adicional de aprendizado. Entretanto, depois que elas forem adotadas e a equipe se tornar experiente, elas podem aumentar significativamente a velocidade do ciclo de vida do desenvolvimento do sistema.

Clareza dos Requisitos do Usuário Quando as exigências do usuário sobre o que o sistema deve fazer não estão claras ou estão sujeitas a alterações, é difícil compreendê-las apenas falando sobre elas e explicando-as com relatórios escritos. Os usuários normalmente precisam interagir com a tecnologia para realmente entender o que o novo sistema pode fazer e como melhor aplicá-lo às suas necessidades. As metodologias de protótipo, protótipo descartável e XP normalmente são mais apropriadas quando as necessidades do usuário não são claras ou são instáveis, porque fornecem protótipos com os quais os usuários podem interagir no início do SDLC.

Familiaridade com a Tecnologia Quando o sistema usa uma tecnologia nova, com a qual os analistas e programadores não estão familiarizados (p. ex., o primeiro projeto de desenvolvimento Web com Java), quanto mais cedo for a aplicação da nova tecnologia no ciclo de vida de desenvolvimento do software maiores as chances de sucesso. Se o sistema é projetado sem alguma familiaridade com a tecnologia de base, os riscos aumentam porque as ferramentas podem não conseguir fazer o que é necessário. O protótipo descartável é particularmente adequado para os casos de falta de familiaridade com a tecnologia, porque estimula explicitamente os desenvolvedores a criarem protótipos do projeto para áreas com altos riscos. O desenvolvimento em fases também é bom, porque cria oportunidades de investigar a tecnologia com algum nível de profundidade antes que o projeto seja concluído. Embora alguém possa achar que a metodologia de protótipo também é apropriada, ela é muito menos porque em geral os protótipos iniciais que são construídos só tocam superficialmente a nova tecnologia. Como de costume, só após vários protótipos e vários meses é que os desenvolvedores descobrem os pontos fracos ou os problemas da nova tecnologia.

Complexidade do Sistema Os sistemas complexos requerem análise e design cuidadosos e detalhados. O protótipo descartável é particularmente adequado para tais análises e designs detalhados, mas o protótipo, não. As metodologias estruturadas tradicionais podem manipular sistemas complexos, mas sem a capacidade de entregar rapidamente os sistemas e protótipos nas mãos do usuário algumas questões importantes podem passar despercebidas. Embora a metodologia em fases permita que os usuários interajam com o sistema no início do processo, temos observado que as equipes de projeto que seguem essa metodologia tendem a dar menos atenção à análise geral do problema o que fariam se estivessem usando outras metodologias.

Habilidade para Desenvolver Sistemas	Metodologias Estruturadas			Metodologias RAD		Metodologias Ágeis
	Cascata	Paralela	Em Fases	Protótipo	Protótipo Descartável	XP
com Requisitos sem Clareza	Pobre	Pobre	Boa	Excelente	Excelente	Excelente
com Tecnologia Não-familiar	Pobre	Pobre	Boa	Pobre	Excelente	Pobre
que são Complexos	Boa	Boa	Boa	Pobre	Excelente	Pobre
que são Confiáveis	Boa	Boa	Boa	Pobre	Excelente	Boa
com um Cronograma Curto	Pobre	Boa	Excelente	Excelente	Boa	Excelente
com Visibilidade de Agenda	Pobre	Pobre	Excelente	Excelente	Boa	Boa

FIGURA 1-9
Critérios para a Seleção de uma Metodologia

Confiabilidade do Sistema A confiabilidade do sistema é normalmente um fator importante no desenvolvimento do sistema. Afinal de contas, quem deseja um sistema não-confiável? Entretanto, a confiabilidade é apenas um fator entre vários. Para algumas aplicações, a confiabilidade é verdadeiramente crítica (p. ex., equipamento médico, sistemas de controle de mísseis), enquanto para outras ela é simplesmente importante (p. ex., jogos, vídeos da Internet). A metodologia do protótipo descartável é a mais apropriada quando a confiabilidade do sistema é uma prioridade alta, porque ela combina as fases detalhadas de análise e projeto com a capacidade que a equipe de projeto tem de testar muitos enfoques diferentes por meio de protótipos de projeto antes de concluir o projeto. A metodologia de protótipo geralmente não é uma boa opção quando a confiabilidade é crítica, porque ela não tem as fases de análise e design criteriosas que são essenciais para sistemas dependentes.

Cronogramas com Prazos Curtos Os projetos que têm prazos muito curtos são mais apropriados para as metodologias RAD e Ágil, porque elas são projetadas para aumentar a velocidade do desenvolvimento. As metodologias de protótipo, desenvolvimento em fases e XP são excelentes opções quando os prazos de entrega são curtos porque elas permitem que a equipe de projeto se ajuste à funcionalidade no sistema com base em uma data de entrega específica; e se o cronograma do projeto começar a atrasar, ela pode ser ajustada removendo-se a funcionalidade da versão ou do protótipo em desenvolvimento. A metodologia de cascata é a pior opção quando o tempo é importante, porque ela não facilita as alterações no cronograma.

Visibilidade de Cronograma Um dos maiores desafios do desenvolvimento de sistemas é saber se um projeto está dentro do cronograma. Isso é particularmente verdadeiro para as metodologias estruturadas, porque o projeto e a implementação ocorrem no final do projeto. As metodologias RAD movem grande parte das decisões de projeto críticas para um ponto anterior no projeto, para ajudar os gerentes de projeto a reconhecer e tratar os fatores de risco e manter as expectativas sob controle.

HABILIDADES E PAPÉIS DA EQUIPE DE PROJETO

Como deve estar claro a partir das várias fases e etapas executadas durante o SDLC, a equipe de projeto precisa de várias habilidades. Os membros do projeto são *agentes de mudança* que identificam as maneiras de melhorar uma empresa, constroem um sistema de informações para suportá-las e treinam e motivam outras pessoas a usar o sistema. Liderar um esforço de mudança empresarial bem-sucedido é uma das tarefas mais difíceis que alguém pode empreender. Entender o que mudar, como mudar e convencer as pessoas da necessidade da mudança requer uma vasta gama de habilidades. Essas habilidades podem ser divididas em seis categorias principais: técnica, empresarial, analítica, social, gerencial e ética.

Os analistas devem ter habilidades técnicas para entender o ambiente técnico existente na empresa, a tecnologia que sustentará o novo sistema e o modo no qual ambos podem ser ajustados em uma solução técnica integrada. As habilidades empresariais são necessárias para compreender como a tecnologia de informações (TI) pode ser aplicada a situações comerciais e para garantir que a TI agregará um valor real. Os analistas são solucionadores de problemas contínuos tanto no nível do projeto quanto no organizacional, e colocam suas habilidades analíticas em teste regularmente.

Com freqüência, os analistas precisam se comunicar pessoalmente, de forma efetiva, com os usuários, com os gerentes da empresa (que quase sempre têm muito pouca experiência com tecnologia) e com os programadores (que têm mais experiência técnica que os analistas). Eles devem ser capazes de fazer apresentações para grupos grandes e pequenos, além de escrever relatórios. Eles precisam não apenas ter fortes habilidades sociais, mas também precisam gerenciar as pessoas com quem trabalham e a pressão e os riscos associados a situações pouco claras.

SUA VEZ **1-1 Selecionando uma Metodologia**

Suponha que você seja um analista da ABC Company, uma grande firma de consultoria com escritórios em todo o mundo. A empresa quer construir um novo sistema de gerenciamento de conhecimentos que possa identificar e controlar a experiência de consultores individuais em qualquer lugar do mundo com base na sua formação e nos vários projetos de consultoria nos quais trabalharam. Admita que isso é uma idéia inovadora, que nunca foi experimentada antes na ABC ou em qualquer outro lugar. A ABC tem uma rede internacional, mas os escritórios em cada país podem usar hardwares e softwares diferentes. A administração da ABC deseja que o sistema esteja pronto e funcionando em um ano.

Pergunta:
Qual a metodologia que você recomendaria para ser usada na ABC Company? Por quê?

Finalmente, os analistas devem lidar de maneira razoável, honesta e ética com os outros membros da equipe de projeto, gerentes e usuários do sistema. Os analistas freqüentemente lidam com informações confidenciais ou com informações que se forem compartilhadas com outras pessoas podem causar prejuízos (p. ex., discordância entre os funcionários); é importante manter a confiança entre todas as pessoas.

Além desses seis conjuntos de habilidades gerais, os analistas precisam ter muitas habilidades específicas que estão associadas aos papéis desempenhados em um projeto. No início do desenvolvimento dos sistemas, a maioria das empresas espera que uma pessoa, o analista, tenha todas as habilidades específicas para conduzir um projeto de desenvolvimento de sistemas. Algumas empresas pequenas ainda esperam que apenas uma pessoa desempenhe muitos papéis, mas como as empresas e a tecnologia têm-se tornado mais complexas, grande parte das empresas grandes cria agora equipes de projeto que dispõem de várias pessoas com responsabilidades claramente definidas. Diferentes empresas dividem os papéis de diferentes maneiras, mas a Figura 1-10 apresenta um conjunto muito usado de papéis da equipe de projeto. A maioria das equipes de SI inclui muitas outras pessoas, como *programadores* que realmente escrevem os programas que fazem parte do sistema e *escritores técnicos*, que preparam as telas de ajuda e outras documentações (como manuais de usuários, manuais de sistemas).

Analista de Negócios

O *analista de negócios* se concentra nas questões empresariais que envolvem o sistema. Essas questões incluem identificar o valor que o sistema agregará, desenvolver idéias e sugestões para o modo

Papel	Responsabilidades
Analista de negócios	Analisar os aspectos de negócios importantes do sistema Identificar como o sistema agregará valor Projetar os novos processos e políticas de negócios
Analista de sistemas	Identificar como a tecnologia pode melhorar os processos de negócios Projetar os novos processos de negócios Projetar o sistema de informações Garantir que o sistema esteja de acordo com os padrões do sistema de informações
Analista de infra-estrutura	Garantir que o sistema esteja de acordo com os padrões de infra-estrutura Identificar as mudanças de infra-estrutura necessárias para suportar o sistema
Analista de gerenciamento de mudança	Desenvolver e executar um plano de gerenciamento de mudança Desenvolver e executar um plano de treinamento de usuário
Gerente de projeto	Gerenciar a equipe de análise, programadores, escritores técnicos e outros especialistas Desenvolver e monitorar o plano de projeto Atribuir recursos Servir como o principal ponto de contato para o projeto

FIGURA 1-10
Papéis da Equipe de Projeto

como os processos da empresa podem ser melhorados e projetar os novos processos e políticas junto com o analista de sistemas. Essa pessoa provavelmente terá experiência administrativa e algum tipo de treinamento profissional (p. ex., o analista de negócios para sistemas de contabilidade provavelmente será um contador formado com registro no CRC — Conselho Regional de Contabilidade). Ele representa os interesses do responsável pelo projeto e os usuários finais do sistema. O analista de negócios ajuda nas fases de planejamento e projeto, mas é mais ativo na fase de análise.

Analista de Sistemas

O *analista de sistemas* se concentra nas questões do sistema de informações que envolvem o sistema. Essa pessoa desenvolve idéias e sugestões para o modo como a tecnologia da informação pode melhorar os processos da empresa, projeta novos processos da empresa com ajuda do analista de negócios, projeta o novo sistema de informações e garante que todos os padrões do SI serão mantidos. Os analistas de sistemas provavelmente terão treinamento e experiência significativos em análise e projeto, programação e ainda em áreas da empresa. Ele representa os interesses do departamento de SI e trabalha intensamente em todo o projeto, mas talvez um pouco menos durante a fase de implementação.

Analista de Infra-estrutura

O *analista de infra-estrutura* se concentra em questões técnicas que envolvem o modo como o sistema vai interagir com a infra-estrutura técnica da empresa (p. ex., hardware, software, redes e bancos de dados). As tarefas do analista de infra-estrutura incluem garantir que o novo sistema de informações esteja de acordo com os padrões organizacionais e identificar as mudanças de infra-estrutura necessárias para suportar o sistema. Esse profissional provavelmente terá treinamento e experiência significativos em rede, administração de bancos de dados e vários produtos de hardware e software. Ele representa os interesses da empresa e do grupo do SI que, por fim, terá de operar e suportar o novo sistema depois que ele for instalado. O analista de infra-estrutura trabalha em todo o projeto, mas talvez um pouco menos durante as fases de planejamento e análise.

Analista de Gerenciamento de Mudança

O *analista de gerenciamento de mudança* se concentra nas questões de pessoal e de gerenciamento que envolvem a instalação do sistema. Os papéis desse profissional incluem garantir que a documentação e o suporte adequado estejam disponíveis para os usuários, fornecendo treinamento do usuário no novo sistema e desenvolvendo estratégias para superar a resistência às mudanças. Esse analista provavelmente terá treinamento e experiência significativos em comportamento organizacional, em geral, e em gerenciamento de mudanças, em particular. Ele representa os interesses do responsável pelo projeto e dos usuários para que o sistema está sendo projetado. O analista de gerenciamento de mudança trabalha mais ativamente durante a fase de implementação, mas começa a assentar a base durante as fases de análise e projeto.

Gerente de Projeto

O *gerente de projeto* é responsável por garantir que o projeto esteja concluído dentro do prazo e dentro do orçamento, e que o sistema proporcione todos os benefícios pretendidos pelo responsá-

SUA VEZ **1-2 Trabalhando como um Analista**

Suponha que você decida se tornar um analista após se formar. Que tipo de analista você preferiria ser? Que tipo de cursos você deveria fazer antes de se formar? Que tipo de estágios ou trabalhos temporários você deveria procurar?

Questão:
Redija um texto curto descrevendo como você se preparará para trabalhar como um analista.

vel pelo projeto. O papel do gerente do projeto inclui gerenciar os membros da equipe, desenvolver o plano do projeto, atribuir recursos e ser o principal ponto de contato quando as pessoas de fora da equipe tiverem perguntas sobre o projeto. Esse profissional provavelmente terá uma experiência significativa em gerenciamento de projeto e terá trabalhado durante muitos anos como um analista de sistemas. Ele representa os interesses do departamento de SI e do responsável pelo projeto. O gerente de projeto trabalha intensamente durante todas as fases do projeto.

RESUMO

O Ciclo de Vida de Desenvolvimento de Sistemas

Todos os projetos de desenvolvimento de sistema seguem essencialmente o mesmo processo básico denominado ciclo de vida de desenvolvimento de sistemas (SDLC, *system development life cycle*). O SDLC começa com uma fase de planejamento, na qual a equipe de projeto identifica o valor agregado do sistema, conduz uma análise de viabilidade e planeja o projeto. A segunda fase é a de análise, na qual a equipe desenvolve uma estratégia de análise, reúne as informações, escreve os casos de uso, constrói um modelo de processo e um modelo de dados. Na próxima fase, a de projeto, a equipe desenvolve os projetos físico, de arquitetura, de interface, de armazenamento de dados e de programa. Na final, da implementação, o sistema é construído, instalado e mantido.

Metodologias de Desenvolvimento de Sistemas

As metodologias de projeto estruturado, como o desenvolvimento em cascata e paralelo, usam uma abordagem passo a passo formal para o SDLC que se move logicamente de uma fase para a seguinte (e dificulta o caminho contrário). Elas produzem um sistema sólido e bem organizado, mas podem ignorar alguns requisitos porque os usuários devem especificá-los no início do processo de projeto, antes de ver o sistema real. As metodologias de desenvolvimento de aplicação rápida (RAD, *rapid application development*) tentam acelerar o desenvolvimento e facilitam para os usuários a especificação dos requisitos, ao desenvolver partes do sistema tanto produzindo versões diferentes (desenvolvimento em etapas) quanto usando protótipos (protótipo, protótipo descartável). As metodologias de desenvolvimento ágil se concentram no aperfeiçoamento do SDLC, eliminando muitas das tarefas e dos prazos associados à definição de requisitos e documentação. A escolha de uma metodologia é influenciada pela clareza dos requisitos do usuário, pela familiaridade com a tecnologia de base, pela complexidade do sistema, pela necessidade de confiabilidade do sistema, pelas pressões de prazo e pela necessidade de ver o progresso no cronograma.

Habilidades e Papéis da Equipe de Projeto

A equipe de projeto precisa de muitas habilidades. Todos os analistas precisam ter habilidades gerais, como técnica, empresarial, analítica, social, gerencial e ética. Entretanto, os diferentes tipos de analistas precisam de habilidades específicas, além dessas gerais. Os analistas de negócios normalmente têm habilidades administrativas que os ajudam a entender as questões administrativas que envolvem o sistema, enquanto os analistas de sistemas têm experiência significativa em análise e projeto além de programação. O analista de infra-estrutura se concentra nas questões técnicas que envolvem o modo como o sistema vai interagir com a infra-estrutura técnica da empresa, e o analista de gerenciamento de mudanças se concentra nas questões de pessoal e gerenciamento que envolvem a instalação do sistema. Além dos analistas, as equipes de projeto incluirão um gerente de projeto, programadores, redatores técnicos e outros especialistas.

TERMOS IMPORTANTES

Agente de mudança
Análise de viabilidade
Analista de gerenciamento de mudança
Analista de infra-estrutura
Analista de negócios
Analista de sistemas
Binder de projeto
Ciclo de vida do desenvolvimento do sistema (SDLC)
Comitê de aprovação
Construção
Conversão
Desenvolvimento ágil
Desenvolvimento de aplicação rápida (RAD)
Desenvolvimento em cascata
Desenvolvimento em fases
Desenvolvimento paralelo
Especificação de sistema
Estratégia de análise
Etapa
Extreme programming (XP)
Fase
Fase de análise
Fase de projeto
Fase de implementação
Fase de planejamento
Gerente de projeto
Metodologia
Metodologia centrada em dados
Metodologia centrada em processos
Metodologia orientada a objeto
Modelo de dados
Modelo de processo
Plano de suporte
Plano de trabalho
Plano de treinamento
Programador
Projeto de armazenamento de dados
Projeto de arquitetura
Projeto de interface
Projeto de programa
Projeto estruturado
Proposta de sistema
Protótipo
Protótipo de projeto
Protótipo de sistema
Protótipo descartável
Redator técnico
Refinamento gradual
Responsável pelo projeto
Resultado
Seleção de projeto
Sistema futuro
Sistema no estado
Solicitação de sistema
Técnica
Versão

PERGUNTAS

1. Compare e constraste as fases, etapas, técnicas e resultados.
2. Descreva as fases principais do ciclo de vida de desenvolvimento de sistemas (SDLC).
3. Descreva as principais etapas da fase de planejamento. Quais são alguns dos resultados importantes?
4. Descreva as principais etapas da fase de análise. Quais são alguns dos resultados importantes?
5. Descreva as principais etapas da fase de projeto. Quais são alguns dos resultados importantes?
6. Descreva as principais etapas da fase de implementação. Quais são alguns dos resultados importantes?
7. O que significa *refinamento gradual* no contexto do SDLC?
8. Compare e contraste as metodologias centradas em processo, as centradas em dados e as orientadas a objetos.
9. Compare e contraste as metodologias de projeto estruturado no geral com as metodologias de desenvolvimento de aplicação rápida (RAD) no geral.
10. Compare e constraste extreme programming e protótipo descartável.
11. Descreva os principais elementos e questões do desenvolvimento em cascata.
12. Descreva os principais elementos e questões do desenvolvimento paralelo.
13. Descreva os principais elementos e as questões do desenvolvimento em fases.
14. Descreva os principais elementos e as questões do protótipo.
15. Quais são os principais fatores na seleção de uma metodologia?
16. Quais são as seis habilidades gerais que toda equipe de projeto deve ter?
17. Quais são os principais papéis em uma equipe de projeto?
18. Compare e constrate o papel de um analista de sistemas, de um analista de negócios e de um analista de infra-estrutura.
19. Por que você acha que a metodologia mais conhecida nos anos 1970 e 1980 foi o desenvolvimento em cascata? Por que você acha que as abordagens do RAD são mais comuns nos dias de hoje? Por que as abordagens Ágeis estão se tornando conhecidas?
20. Que fase do SDLC é a mais importante?

EXERCÍCIOS

A. Suponha que você seja um gerente de projetos usando a metodologia de desenvolvimento em cascata em um projeto grande e complexo. Seu gerente acabou de ler o último artigo da *Computerworld*, que advoga a substituição da metodologia em cascata pelo protótipo, e chega no escritório solicitando que você faça a troca. O que você diz?

B. As seis metodologias básicas discutidas neste capítulo podem ser combinadas e integradas para formar novas metodologias híbridas. Suponha que você fosse combinar o protótipo descartável com o uso do desenvolvimento paralelo. Com o que a metodologia se pareceria? Desenhe uma figura (similar à Figura 1-7). Como essa nova tecnologia poderia ser comparada às outras? Desenvolva uma nova coluna para a Figura 1-8.

C. Suponha que você seja um analista trabalhando para uma empresa pequena no sentido de desenvolver um sistema de contabilidade. Que metodologia você deveria usar? Por quê?

D. Suponha que você seja um analista desenvolvendo um novo sistema de informações executivas (EIS, *executive information system*) cujo objetivo seria fornecer informações estratégicas importantes dos bancos de dados corporativos existentes para os executivos seniores, a fim de ajudar nas suas tomadas de decisão. Que metodologia você usaria? Por quê?

E. Suponha que você seja um analista desenvolvendo um novo sistema de informações para automatizar as transações de vendas e gerenciamento de estoque de cada loja de uma grande rede de lojas. O sistema seria instalado em cada loja e os dados trocados com um computador *mainframe* no escritório sede da empresa. Que metodologia você usaria? Por quê?

F. Observe a seção de classificados do seu jornal. Que tipos de oportunidades de trabalho estão disponíveis para pessoas que desejam posições de analista? Compare e contraste as habilidades que os anúncios solicitam com as habilidades que apresentamos neste capítulo.

G. Pense sobre sua posição de analista ideal. Escreva um anúncio de jornal para contratar alguém para essa posição. Que requisitos o cargo teria? Que habilidades e experiência seriam necessárias? Como os currículos poderiam demonstrar que os candidatos têm as habilidades e a experiência apropriadas?

MINICASOS

1. Barbara Singleton, gerente de vendas regionais da WAMAP Company, solicitou que o departamento de SI desenvolvesse um sistema de gerenciamento e controle da força de vendas que pudesse permitir um melhor monitoramento do desempenho da sua equipe de vendas. Infelizmente, devido à grande sobrecarga de trabalho que o departamento de SI enfrenta, foi dada uma baixa prioridade à sua solicitação. Após seis meses sem que o departamento de SI desse algum retorno, Barbara decidiu resolver o assunto sozinha. Com base nos conselhos de amigos, Barbara comprou um PC e um software de banco de dados simples e construiu sozinha um sistema de gerenciamento e controle da força de vendas.

   Embora o sistema de Barbara tenha sido "concluído" em cerca de seis meses, ele ainda tem muitos recursos que não funcionam corretamente, e algumas funções estão cheias de erros. A assistente de Barbara estava tão desconfiada do sistema que voltou a usar secretamente seu velho sistema baseado em papéis, visto que ele era muito mais confiável.

   Durante um jantar com um amigo analista de sistemas, Bárbara se queixou: "Não sei o está errado com esse projeto. Ele me parecia tão simples. Aquele pessoal do departamento de SI quis que eu seguisse esse conjunto elaborado de etapas e tarefas, mas eu não acho que tudo isso realmente se aplique a um sistema baseado em PC. Apenas achei que poderia construir esse sistema e ajustá-lo até conseguir o que queria sem toda essa confusão de metodologia que o pessoal do SI está me empurrando. Isso não se aplica apenas aos seus sistemas maiores e mais caros?"

   Admitindo que você é o amigo analista de Bárbara, como responderia à sua queixa?

2. Marcus Weber, o gerente de projeto da ICAN Mutual Insurance Co., está revendo o planejamento da equipe para seu próximo projeto importante, o desenvolvimento de uma assistência a agentes de seguros baseada em sistema pericial. Esse novo sistema envolverá uma maneira totalmente nova de os agentes de seguros executarem suas tarefas. O sistema de assistência a agentes de seguros funcionará como um tipo de supervisor de seguros, revendo os principais elementos de cada aplicação, verificando a consistência nas decisões de cada agente e garantindo que nenhum fator crítico será ignorado. O objetivo do novo sistema é melhorar a qualidade das decisões do agente e sua produtividade. Espera-se que o novo sistema mude substancialmente o modo como a equipe de agentes de seguros faz seu trabalho.

   Marcus desanimou ao saber que devido às restrições de orçamento, deverá escolher um dos dois membros disponíveis da equipe. Barry Filmore teve experiência e treinamento considerável em comportamento individual e organizacional. Barry trabalhou em vários outros projetos nos quais os usuários finais precisavam fazer ajustes significativos ao novo sistema e ele parece ter o dom de antecipar os problemas e suavizar a transição para um novo ambiente de trabalho. Marcus esperava ter o envolvimento de Barry nesse projeto.

   Outro membro potencial da equipe de Marcus é Kim Danville. Antes de trabalhar na ICAN Mutual, Kim teve uma considerável experiência de trabalho com as tecnologias de sistemas periciais que a ICAN escolheu para seu projeto de sistema pericial. Marcus estava contando com Kim para ajudar a integrar a nova tecnologia de sistema pericial no ambiente de sistemas da ICAN e também fornecer treinamento e *insights* sobre o trabalho para outros desenvolvedores dessa equipe.

   Dado que o orçamento de Marcus só permitirá que Kim ou Barry faça parte do projeto e não ambos, que opção você recomendaria para ele? Justifique sua resposta.