Aplicando Gerenciamento de Requisitos com Casos de Uso

Roger Oberg, Leslee Probasco e Maria Ericsson

Rational Software White Paper

TP 505 (Versão 1.4)

Rational®
the software development company

Índice Analítico

Se você for iniciante ou de alguma forma estiver familiarizado com o gerenciamento de requisitos e estiver interessado no aprimoramento do processo de requisitos, este documento oferece uma estrutura com a qual irá desenvolver a sua própria abordagem.

Casos de Uso e SRS (Especificações de Requisitos de Software)

Para tornar as atividades de gerenciamento de requisitos mais significativas aos leitores, os autores escolheram tipos de documentos específicos e outros artefatos de gerenciamento de requisitos do Unified Process da Rational Software e do Idioma Unificado de Modelagem padrão de mercado, que recomendam um processo de engenharia de software conduzido por caso de uso.

Portanto, uma abordagem conduzida por caso de uso para especificar requisitos de software está descrita aqui. Essas atividades também podem ser utilizadas com as Especificações de Requisitos de Software mais tradicionais (por exemplo, padrões IEEE) no lugar dos modelos de caso de uso e de casos de uso ou em adição a eles.

Software e Desenvolvimento de Software no Período do Processo

Para a maioria das equipes de desenvolvimento de software e de sistema, a década de 90 foi intensa em processos quando comparada aos dias em que as coisas eram mais livres e improvisadas. Os padrões para medir e certificar o processo efetivo de desenvolvimento de software foram introduzidos e popularizados. Foram publicados muitos livros e artigos sobre o processo de desenvolvimento de software e material relacionado sobre modelagem e reengenharia de processos de negócios. O número crescente de ferramentas de software ajudou a definir e a aplicar o processo efetivo de desenvolvimento de software. A dependência de software da economia global foi acelerada durante a década, permitindo processos de desenvolvimento e aprimorando a qualidade dos sistemas.

Então, como explicamos a alta incidência de falhas em projetos de software nos dias de hoje? Por que há vários projetos de software, se não a maioria, que ainda sofrem com atrasos, orçamentos excessivos e problemas de qualidade? Como podemos aprimorar a qualidade dos sistemas que construímos conforme nossas empresas, economias nacionais e atividades diárias tornam-se cada vez mais dependentes deles?

As respostas, como sempre, estão nas pessoas, nas ferramentas e nos processos aplicados a nossa profissão. O gerenciamento de requisitos é freqüentemente proposto como uma solução para os problemas contínuos de desenvolvimento de software, no entanto, relativamente pouca atenção tem sido focada no aprimoramento da prática dessa disciplina.

Este documento apresenta os elementos de um processo efetivo de gerenciamento de requisitos e destaca alguns dos obstáculos para sua implementação bem-sucedida.

O gerenciamento de requisitos aplica-se igualmente aos projetos de software e aos projetos nos quais o software é apenas uma parte do resultado final ou não está de modo algum incluído. Por questão de conveniência, este documento daqui em diante utilizará o termo "sistema" para referir-se a todas as essas coisas. No entanto, é a natureza abstrata do desenvolvimento de software, sozinha ou em conjunto com o hardware, que complica o gerenciamento de requisitos e este é o foco principal do documento.

Por Que Gerenciar Requisitos?

Em poucas palavras, equipes de desenvolvimento de software que gerenciam requisitos o fazem porque desejam que seus projetos tenham êxito. Atender aos seus requisitos de projeto define o sucesso. A falha no gerenciamento de requisitos reduz a probabilidade de atender a esses objetivos.

Evidência recente é sustentada:

- Os Relatórios CHAOS da Standish Group, de 1994 e 1997, estabeleceram que os contribuidores mais significativos para a falha de um projeto estão relacionados aos requisitos. [1]

Field Code Changed

- Em dezembro de 1997, a Computer Industry Daily relatou um estudo da Sequent Computer Systems, Inc. com 500 gerentes de TI nos Estados Unidos e no Reino Unido que apontou que 76 por cento dos entrevistados já tinham tido falhas em projetos inteiros durante suas carreiras. A causa mais freqüentemente apontada como falha nos projetos foi "alteração de requisitos do usuário". [2]

Field Code Changed

Evitar falhas deve ser uma motivação suficiente para gerenciar requisitos. Aumentar a probabilidade de um projeto bem-sucedido e de outros benefícios provenientes do gerenciamento de requisitos pode ser igualmente uma motivação. O relatório CHAOS do Standish Group estabeleceu ainda que o bom gerenciamento de requisitos era o fator mais relacionado aos projetos bem-sucedidos.

O Que é um Requisito?

A primeira etapa para entender o gerenciamento de requisitos é concordar com um vocabulário comum. A Rational define um requisito como "uma condição ou capacidade com a qual o sistema [sendo construído] deve estar em conformidade". O IEEE (Institute of Electronics and Electrical Engineers) utiliza uma definição similar.

Os já conhecidos autores de engenharia de requisitos, Merlin Dorfman e Richard H. Thayer, oferecem uma definição compatível e mais refinada que é específica—mas não está necessariamente limitada—ao software.

"Um requisito de software pode ser definido como:

- Uma capacidade de software necessária pelo usuário para resolver um problema atingir um objetivo.
- Uma capacidade de software que deve ser atendida ou que deve ser de posse de um sistema ou componente de sistema para satisfazer um contrato, padrão, especificação ou outra documentação formalmente imposta". [3]

Field Code Changed

O Que É Gerenciamento de Requisitos?

Como requisitos são coisas com as quais o sistema que está sendo construído deve estar em conformidade e a conformidade com alguns conjuntos de requisitos define o sucesso ou a falha de projetos, faz sentido descobrir quais são os requisitos, anotá-los, organizá-los e rastreá-los, caso sejam alterados.

Em outras palavras, o gerenciamento de requisitos é:

- uma abordagem sistemática de fazer levantamento, organizar e documentar os requisitos do sistema e
- um processo que estabelece e mantém concordância entre o cliente e a equipe do projeto na alteração de requisitos do sistema.

Essa definição é similar a de Dorfman e Thayer e à definição do IEEE de "engenharia de requisitos de software". A engenharia de requisitos inclui levantamento, análise, especificação, verificação e gerenciamento dos requisitos de software, em que o "gerenciamento de requisitos de software" é o planejamento e o controle de todas essas atividades relacionadas[4]. Todas essas atividades são incorporadas na definição de gerenciamento de requisitos apresentada aqui e ensinada pela Rational Software. A diferença está principalmente na escolha da palavra "gerenciamento" em vez de "engenharia". O gerenciamento é uma descrição mais apropriada de todas as atividades envolvidas e enfatiza precisamente a importância de se rastrear alterações para manter a concordância entre os envolvidos e a equipe do projeto.

Field Code Changed

Para aqueles que não estão familiarizados com o termo "levantamento", ele é definido como o conjunto de atividades que as equipes empregam para fazer o levantamento ou descobrir pedidos dos envolvidos, determinar as necessidades reais por trás dos pedidos e chegar a um conjunto adequado de requisitos que poderá ser colocado no sistema para atender a essas necessidades.

Os Problemas do Gerenciamento de Requisitos

Portanto, o que poderá ser difícil sobre um processo que pretende assegurar que um sistema esteja em conformidade com as expectativas depositadas nele? Quando colocado na prática em projetos reais, as dificuldades vêm à tona. A Figura 1 exibe os resultados de uma pesquisa de opinião de 1996 realizada com desenvolvedores, gerentes e pessoal de garantia de qualidade.

Ela mostra a porcentagem de entrevistados que tiveram os problemas relacionados a requisitos mais freqüentemente mencionados.

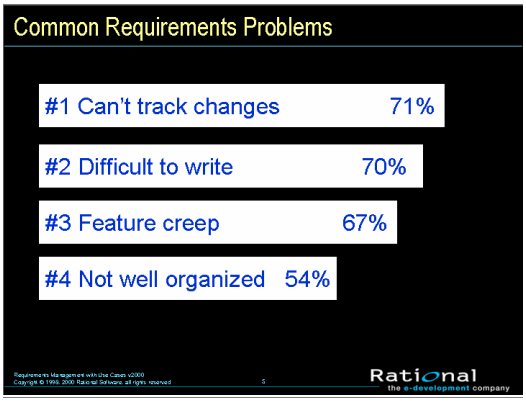

Figura 1—Problemas Comuns de Requisitos

Uma lista mais abrangente de problemas inclui:

- Os requisitos nem sempre são óbvios e têm muitas fontes.
- Nem sempre é fácil expressar os requisitos claramente em palavras.
- Existem diversos tipos de requisitos em diferentes níveis de detalhe.
- O número de requisitos poderá impossibilitar a gerência se não for controlado.
- Os requisitos estão relacionados uns com os outros e com outros distribuíveis do processo de várias maneiras.
- Os requisitos têm propriedades exclusivas ou valores de propriedade. Por exemplo, eles não são igualmente importantes nem igualmente fáceis de atender.
- Há vários envolvidos e responsáveis, o que significa que os requisitos precisam ser gerenciados por grupos de pessoas de diferentes funções.
- Os requisitos são alterados.
- Os requisitos podem ser sensíveis à hora.

Quando esses problemas são combinados com o gerenciamento de requisitos e habilidades do processo inadequados e a falta de ferramentas fáceis de utilizar, muitas equipes perdem a esperança de que poderão gerenciar bem requisitos. A Rational Software desenvolveu a habilidade para instruir equipes nas técnicas e processos de gerenciamento de requisitos. Além disso, o Rational RequisitePro® é uma ferramenta acessível para automatizar o gerenciamento efetivo de requisitos.

Habilidades de Gerenciamento de Requisitos

Para resolver os problemas mencionados acima, a Rational incentiva o desenvolvimento de habilidades-chave. Estas habilidades são apresentadas a seguir e parecem estar em ordem seqüencial, no entanto, em um processo efetivo de gerenciamento de requisitos, elas são aplicadas continuamente em uma ordem variada. Elas são apresentadas aqui na seqüência em que provavelmente você aplicaria à primeira iteração de um projeto novo.

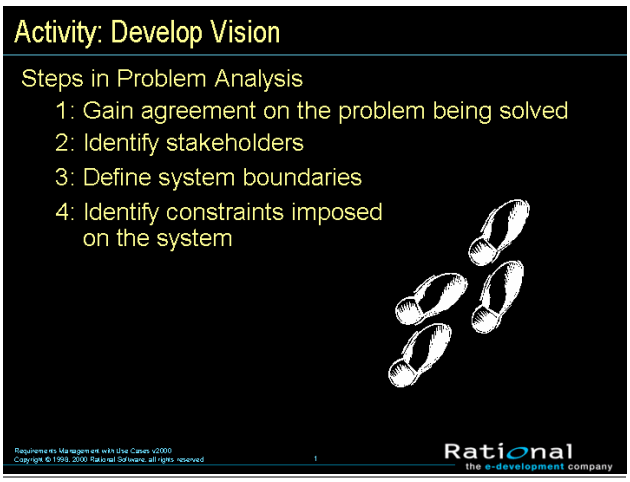

Figura 2—Etapas na Análise de Problemas

Habilidade-chave 1: Analisar o Problema

A análise do problema é feita para compreender os problemas de negócios, as necessidades-alvo iniciais dos envolvidos e propor soluções de nível alto. Essas linhas de raciocínio e análise localizam "o problema por trás do problema".

Durante a análise do problema, chega-se a um acordo em relatório dos problemas reais e os envolvidos são identificados. Os limites e restrições iniciais da solução são definidos a partir das perspectivas técnicas e comerciais. Se apropriado, o caso de negócios para o projeto analisa o retorno do investimento esperado do sistema.

Habilidade-chave 2: Entender as Necessidades dos Envolvidos

Os requisitos têm muitas fontes. Eles podem vir de qualquer um que tenha interesse no resultado do projeto. Os clientes, parceiros, usuários finais e especialistas em domínios são algumas fontes de requisitos. Gerenciamento, membros da equipe do projeto, políticas de negócios e agências reguladoras podem ser outras.

É importante saber como determinar quais devem ser as fontes, como obter acesso a essas fontes e qual a melhor forma de fazer o levantamento de informações delas. Os indivíduos que constituem as fontes primárias para essas informações são conhecidos como os "envolvidos" no projeto.

Se você estiver desenvolvendo um sistema de informações para ser utilizado internamente na sua empresa, poderá incluir na sua equipe de desenvolvimento pessoas com experiência de usuário final e com conhecimento do domínio de negócios. Com bastante freqüência, você começará os debates no nível de um modelo de negócio em vez de no nível de um sistema. Se você estiver desenvolvendo um produto para ser vendido para um estabelecimento comercial, poderá fazer uso extensivo do pessoal de marketing para entender melhor as necessidades dos clientes desse mercado.

As técnicas para o levantamento de requisitos incluem entrevistas, discussão de idéias, protótipos conceituais, questionários e análise competitiva. O resultado do levantamento de requisitos é uma lista dos pedidos ou das necessidades que foram descritos textual e graficamente e que receberam prioridade em relação ao outro.

Habilidade-chave 3: Definir o Sistema

Definir o sistema significa traduzir e organizar as necessidades dos investidores em descrições significativas do sistema a ser construído. No início da definição do sistema, as decisões são tomadas sobre o que constitui um requisito, o formato de

documentação, a formalidade da linguagem, o grau dos requisitos, a prioridade dos pedidos e o esforço estimado, os riscos técnicos e de gerenciamento e o escopo. Parte dessa atividade pode incluir modelos de design e protótipos iniciais diretamente relacionados aos pedidos mais importantes dos investidores.

Utilizamos a palavra "descrição" em vez de "documento" para evitar a limitação percebida inerente no uso comum da última. Uma descrição pode ser um documento escrito, um arquivo eletrônico, uma imagem ou qualquer outra representação utilizada para comunicar requisitos do sistema, com exceção do próprio sistema.

O resultado da definição do sistema é uma descrição do sistema que esteja em idioma natural e também seja gráfica. Alguns formatos sugeridos para a descrição são fornecidos em seções posteriores.

Princípio 55: Escrever em uma Linguagem Natural Antes de um Modelo Mais Formal

Se você escrever o modelo formal primeiro, a tendência será escrever em uma linguagem natural que descreve o modelo em vez do sistema de solução. Considere os seguintes exemplos:

> Para fazer uma chamada de longa distância, o usuário deve levantar o gancho. O sistema deve responder com um tom de discagem. O usuário deve discar "9". O sistema deve responder com um tom de discagem distintivo…
>
> O sistema consiste em quatro estados: Inativo, Tom de Discagem, Tom Discagem Distintivo e Conectado. Para ir do estado inativo para o estado de tom de discagem, levante o gancho. Para ir do estado de tom de discagem para o estado de tom de discagem distintivo, disque "9".

Observe que no último exemplo, o texto não ajuda o leitor.

—Alan M. Davis, 201 Principles of Software Development, 1995

Habilidade-chave 4: Gerenciar o Escopo do Sistema

O escopo de um projeto é definido pelo conjunto de requisitos alocados para ele. O gerenciamento do escopo do projeto para se adequar aos recursos disponíveis (tempo, pessoas e dinheiro) é primordial para o gerenciamento de projetos com êxito. Gerenciar o escopo é uma atividade contínua que requer desenvolvimento iterativo ou incremental, que quebra o escopo do projeto em partes menores mais simples de gerenciar.

O uso de atributos de requisitos, como prioridade, esforço e risco como a base para a negociação da inclusão de um requisito é uma técnica particularmente útil para gerenciar o escopo. Focar os atributos em vez de os requisitos propriamente ditos ajuda a desestimular negociações que são normalmente controversas.

É proveitoso que os líderes de equipe sejam treinados para negociações, e que o projeto tenha um defensor na organização, assim como no lado do cliente. Os defensores do produto ou do projeto devem ter o poder organizacional para recusar alterações de escopo além dos recursos disponíveis ou para expandir recursos a fim de acomodar escopo adicional.

Habilidade-chave 5: Refinar a Definição do Sistema

Com uma definição do sistema de nível alto aceita e um escopo inicial completamente entendido, é possível e econômico investir recursos em definições do sistema mais refinadas. Refinar a definição do sistema inclui duas considerações fundamentais: desenvolver descrições mais detalhadas da definição do sistema de nível alto e verificar se o sistema atenderá as necessidades dos envolvidos e se comportará conforme o descrito.

As descrições são freqüentemente os materiais de referência crítica para as equipes do projeto. As descrições são melhores feitas com o público em mente. Um engano comum é representar o que é complexo para construir com uma definição complexa, particularmente quando o público não pode ou não deseja investir a opinião crítica necessária para obter concordância. Isso leva a dificuldades para explicar a finalidade do sistema para pessoas que estão dentro e fora da equipe do projeto. Em vez disso, você poderá descobrir a necessidade de produzir diferentes tipos de descrições para públicos

diferentes. Este documento inclui formatos sugeridos para linguagem natural detalhada, texto formal e descrições gráficas. Depois de estabelecer o formato da descrição, o refinamento continua em todo o ciclo de vida do projeto.

Habilidade-chave 6: Gerenciar a Alteração de Requisitos

Não importa o cuidado com que você define seus requisitos, eles serão alterados. De fato, alguma alteração de requisitos é desejável! Significa que sua equipe está atraindo os envolvidos. Acomodar a alteração de requisitos é uma medida de sensibilidade e de flexibilidade operacional dos envolvidos da equipe—atributos de equipe que contribuem para projetos bem-sucedidos. A alteração não é o inimigo; mas sim a alteração não gerenciada.

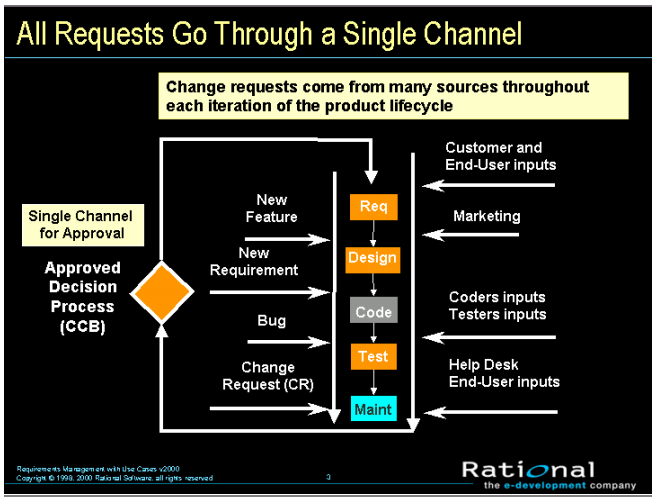

Figura 3—Um Processo para Gerenciar Alterações

Um requisito alterado significa que mais ou menos tempo precisa ser gasto na implementação de um determinado recurso e que uma alteração em um requisito poderá ter impacto em outros requisitos. O gerenciamento de alterações em requisitos inclui atividades como estabelecer uma linha de base, acompanhar o histórico de cada requisito, determinar quais dependências são importantes para o rastreio, estabelecer relacionamentos rastreáveis entre itens relacionados e manter o controle de versão. Conforme a Figura 3 ilustra, também é importante estabelecer um controle de alterações ou processo de aprovação, o que requer que todas as alterações propostas sejam revisadas pelos membros da equipe designada. Às vezes, esse único canal de controle de alterações é chamado de CCB (Conselho de Controle de Alterações).

Conceitos de Requisitos Importantes

Para aplicar habilidades de gerenciamento de requisitos a um projeto, determinados conceitos de gerenciamento de requisitos são úteis para que cada um no projeto entenda. Eles incluem:

- Tipos de requisitos
- Equipes com diferentes funções
- Rastreabilidade
- Atributos multidimensionais
- Histórico de alterações

Tipos de Requisitos

Quanto maior e mais complexo o sistema, mais tipos de requisitos aparecerão. Um tipo de requisito é simplesmente uma classe de requisitos. Ao identificar os tipos de requisitos, as equipes podem organizar números grandes de requisitos em grupos significativos e mais gerenciáveis. O estabelecimento de tipos diferentes de requisitos em um projeto ajuda os membros da equipe a classificarem pedidos de alterações e a se comunicarem mais claramente.

Normalmente, um tipo de requisito pode ser dividido ou decomposto em outros tipos. As regras de negócios e as declarações de visão podem ser tipos de requisitos de nível alto dos quais as equipes derivam as necessidades dos usuários, os recursos e os tipos de requisitos do produto. Os casos de uso e outras formas de modelagem conduzem os requisitos de design que podem ser decompostos para requisitos de software e representados em modelos de análise e design. Os requisitos de teste são derivados dos requisitos de software e decompostos em procedimentos de teste específicos. Quando há centenas, milhares ou mesmo dezenas de milhares de instâncias de requisitos em um determinado projeto, a classificação de requisitos em tipos torna o projeto mais gerenciável.

Equipes de Diferentes Funções

Ao contrário de outros processos, como teste ou modelagem de aplicativo, que podem ser gerenciados dentro de um único grupo de negócios, o gerenciamento de requisitos deve envolver qualquer um que possa contribuir com seu conhecimento no processo de desenvolvimento. Ele deve incluir as pessoas que representam as expectativas do cliente e da empresa. Gerentes de desenvolvimento, gerentes de produtos, analistas, engenheiros de sistema e até mesmo clientes devem participar. As equipes de requisitos devem incluir também aqueles que criam a solução do sistema—engenheiros, arquitetos, designers, programadores, escritores técnicos e outros contribuidores técnicos. Testadores e outro pessoal de QA (Garantia de Qualidade) devem ser contados como membros importantes da equipe.

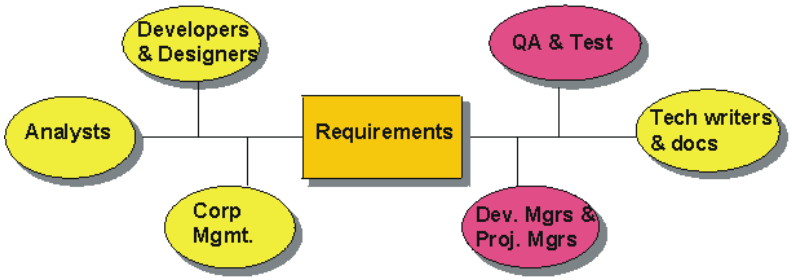

Figura 4—Equipe de Requisitos de Diferentes Funções

Freqüentemente, a responsabilidade por autorizar e manter um tipo de requisito pode ser alocada pela área funcional, contribuindo adicionalmente para um melhor gerenciamento de projetos grandes. A natureza do gerenciamento de requisitos em diferentes funções é uma dos aspectos mais desafiadores da disciplina.

Rastreabilidade

Conforme implicado na descrição de tipos de requisito, nenhuma expressão individual de um requisito fica sozinha. Os pedidos dos envolvidos estão relacionados aos recursos do produto propostos para atendê-los. Os recursos dos produtos estão relacionados a requisitos individuais que especificam os recursos em termos de comportamento funcional e não funcional. Os casos de teste estão relacionados aos requisitos que eles verificam e validam. Os requisitos podem ser dependentes de outros requisitos ou podem ser mutuamente exclusivos. Para que as equipes determinem o impacto das alterações e sintam-se seguras que o sistema está em conformidade com as expectativas, esses relacionamentos de rastreabilidade devem ser entendidos, documentados e mantidos. Enquanto a rastreabilidade é um dos conceitos mais difíceis de implementar no gerenciamento de requisitos, ela é essencial para acomodar alterações. O estabelecimento de tipos de requisitos claros e a incorporação da participação de diferentes funções podem tornar a implementação e a manutenção da rastreabilidade mais fáceis. Para obter informações adicionais sobre estratégias diferentes para rastreabilidade de requisitos, consulte o white paper, "Estratégias de Rastreabilidade para Gerenciar Requisitos com Casos de Uso" [5].

Field Code Changed

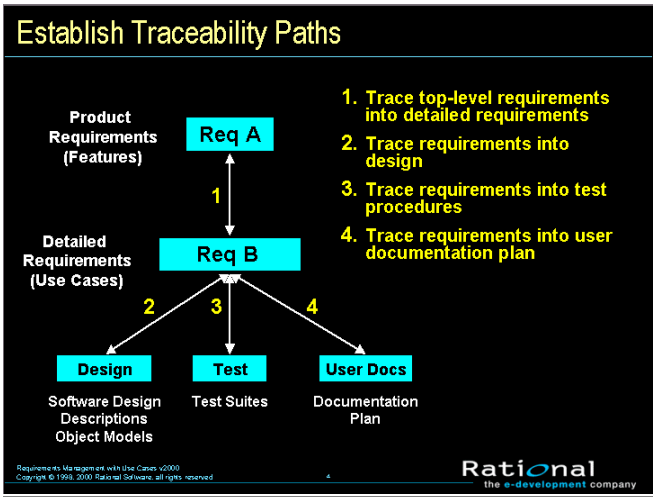

Figura 5—Rastreabilidade de Requisitos

Atributos Multidimensionais

Cada tipo de requisito tem atributos e cada requisito individual tem valores de atributo diferentes. Por exemplo, os requisitos podem receber prioridades, ser identificados por origem e base racional, delegados a subequipes específicas dentro de uma área funcional, receber uma designação de grau de dificuldade ou associados a uma iteração particular do sistema. Para ilustrar, a Figura 6 exibe atributos para um Tipo de Requisito de Recurso de um Projeto de Aprendizado na ferramenta de gerenciamento de requisitos Rational RequisitePro. Conforme implicado pelo título da tela, o tipo de requisito e os atributos para cada tipo são definidos para o projeto inteiro, assegurando consistência de uso entre a equipe.

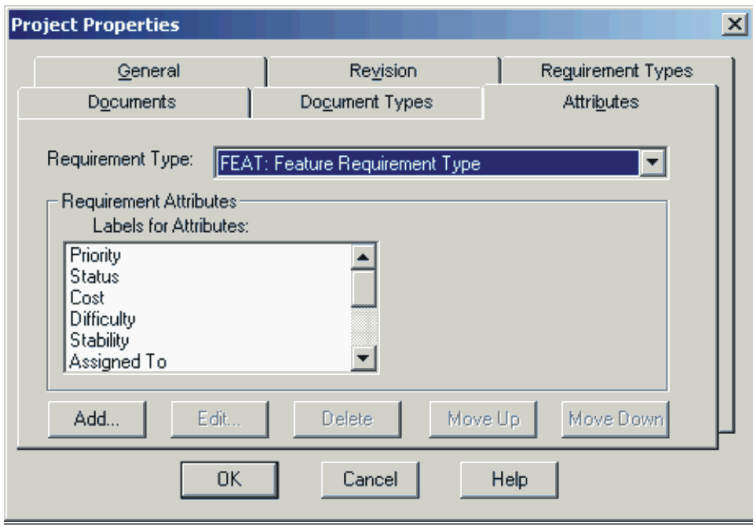

Figura 6—Definindo Atributos de Requisito para Recursos

Na Figura 7, as instâncias dos requisitos de recurso são exibidas para um projeto específico no RequisitePro. Observe que mesmo sem exibir o texto inteiro para cada requisito, podemos aprender muito sobre cada requisito a partir de seus valores de atributo. Nesse caso, sua prioridade e dificuldade—sem dúvida designadas por membros da equipe diferentes—ajudarão a equipe a começar o escopo do projeto para recursos e horas disponíveis, levando em conta as prioridades dos envolvidos e uma estimativa aproximada do esforço refletido no valor do atributo de dificuldade. Em tipos de requisitos mais detalhados, os atributos de prioridade e esforço podem ter valores mais específicos, por exemplo, tempo estimado, linhas de código e assim por diante) com os quais irá refinar mais o escopo. Esse aspecto multidimensional de um requisito, composto por tipos diferentes de requisitos—cada um com seus próprios atributos—é essencial para organizar grandes números de requisitos e para gerenciar o escopo global do projeto.

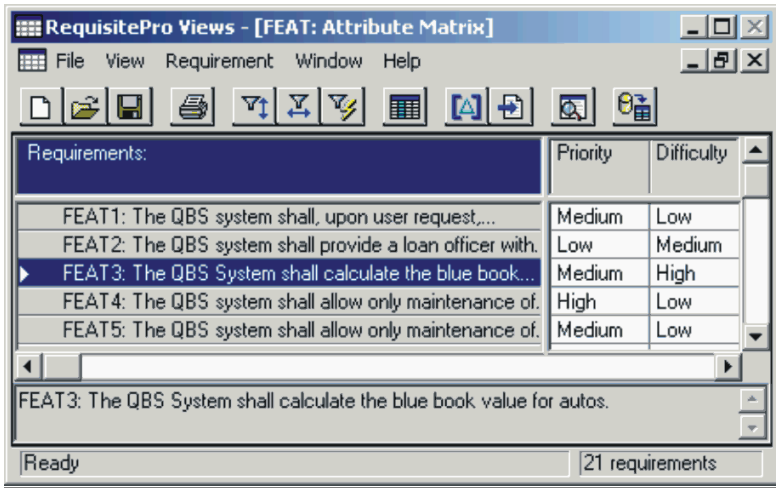

Figura 7—Definindo Valores de Atributo para Cada Requisito

Histórico de Alterações

Requisitos individuais e uma coleta de requisitos têm histórias que se tornam significativas com o tempo. A alteração é inevitável e desejável para acompanhar uma ambiente de alterações e a evolução da tecnologia. O registro das versões de requisitos do projeto permitem que os líderes de equipe capturem as razões para alterar o projeto, como um novo release do sistema. A compreensão de que uma coleta de requisitos pode estar associada a uma versão específica de software permite gerenciar alterações incrementalmente, reduzindo o risco e aprimorando a probabilidade de atender marcos. Como os requisitos individuais evoluem, é importante entender seu histórico: o que mudou, por que, quando e quem autorizou.

Colocando o Gerenciamento de Requisitos para Trabalhar

O gerenciamento de requisitos emprega habilidades-chave e conceitos apresentados acima para identificar e resolver os problemas com êxito.

Para construir um sistema que realmente atenda às necessidades dos clientes, a equipe do projeto deve primeiro definir o problema a ser resolvido pelo sistema. Em seguida, a equipe deve identificar os envolvidos a partir dos quais são feitos o levantamento, a descrição e a priorização das necessidades da empresa e do usuário. A partir desse conjunto de expectativas ou necessidades de nível alto, um conjunto de recursos do produto ou do sistema deve ser aceito.

Os requisitos de software detalhados precisam ser escritos de uma forma que sejam entendido pelos clientes e pela equipe de desenvolvimento. Achamos que utilizar a linguagem do cliente para descrever esses requisitos de software é mais efetivo na obtenção de entendimento e aceitação. Esses requisitos de software detalhados são então utilizados como entrada para as especificações de design do sistema, bem como para planos e procedimentos de teste necessários para implementação e validação. Os requisitos de software devem também conduzir o planejamento e o design iniciais da documentação do usuário.

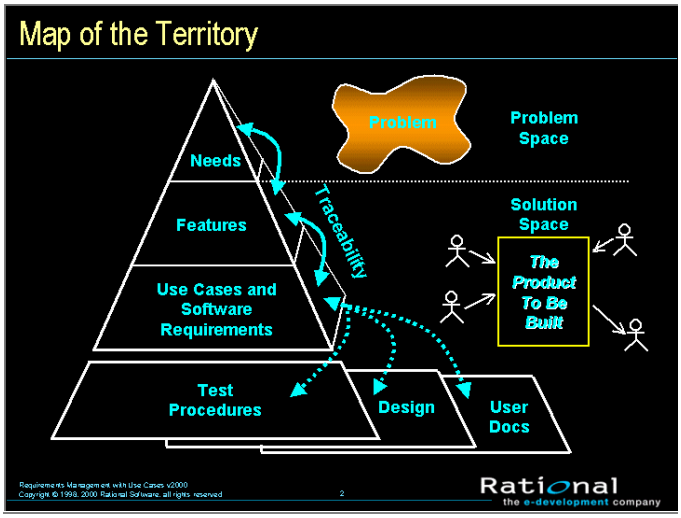

Figura 8—Visão Geral da Estrutura de Gerenciamento de Requisitos

Para facilitar isso, a equipe do projeto deve:

- Concordar com um vocabulário comum para o projeto.
- Desenvolver uma visão do sistema que descreve o problema a ser resolvido pelo sistema, bem como de seus recursos principais.

- Fazer o levantamento das necessidades dos envolvidos em pelo menos cinco áreas importantes: funcionalidade, usabilidade, confiabilidade, desempenho e suportabilidade.
- Determinar quais tipos de requisitos utilizar.
- Selecionar atributos e valores para cada tipo de requisito.
- Escolher os formatos nos quais os requisitos são descritos.
- Identificar membros da equipe que serão autores, contribuirão ou simplesmente visualizarão um ou mais tipos de requisitos.
- Decidir qual rastreabilidade é necessária.
- Estabelecer um procedimento para propor, revisar e resolver alterações em requisitos.
- Desenvolver um mecanismo para rastrear o histórico de requisitos.
- Criar relatórios de progresso e status para membros da equipe e gerenciamento.

Essas atividades essenciais de gerenciamento de requisitos são independentes de indústria, metodologia de desenvolvimento ou ferramentas de requisitos. Elas também são flexíveis, o que permite o gerenciamento efetivo de requisitos em ambientes de desenvolvimento de aplicativos mais rigorosos e mais rápidos.

Algumas Informações sobre Documentos

A decisão de descrever requisitos em documentos merece alguma atenção. Por um lado, a escrita é uma forma amplamente aceita de comunicação e, para a maioria das pessoas, uma coisa natural de se fazer. Por outro lado, o objetivo do projeto é produzir um sistema e não documentos.

O senso comum e a experiência ensinam que a decisão não é se irá documentar requisitos, mas como irá documentá-los. Gabaritos de documentos fornecem um formato consistente para gerenciamento de requisitos. O Rational RequisitePro oferece esses gabaritos e o recurso adicional de vincular requisitos dentro de um documento a um banco de dados que contém todos os requisitos do projeto. Esse recurso exclusivo permite que os requisitos sejam documentados naturalmente enquanto se tornam mais acessíveis e gerenciáveis em um banco de dados relacional.

Gerenciamento de Requisitos: Atividades

O gerenciamento de requisitos pode seguir um número infinito de caminhos específicos do domínio. A seguinte abordagem prescreve seis atividades detalhadas que se aplicam a cada uma das habilidades-chave de gerenciamento de requisitos, mas podem ser aplicadas a qualquer domínio.

Os seguintes diagramas de atividades são do Rational Unified Process [6] Disciplina de Requisitos. Essas atividades são expressas em termos de funções, tarefas e artefatos (entrada ou saída) conforme resumido na Figura 9. O texto associado neste documento descreve cada atividade brevemente, na esperança de estimular suas idéias e interesses no aprimoramento do processo de gerenciamento de requisitos. Informações adicionais sobre o Rational Unified Process podem ser localizadas em www.ibm.com/software/rational.

Field Code Changed

Field Code Changed

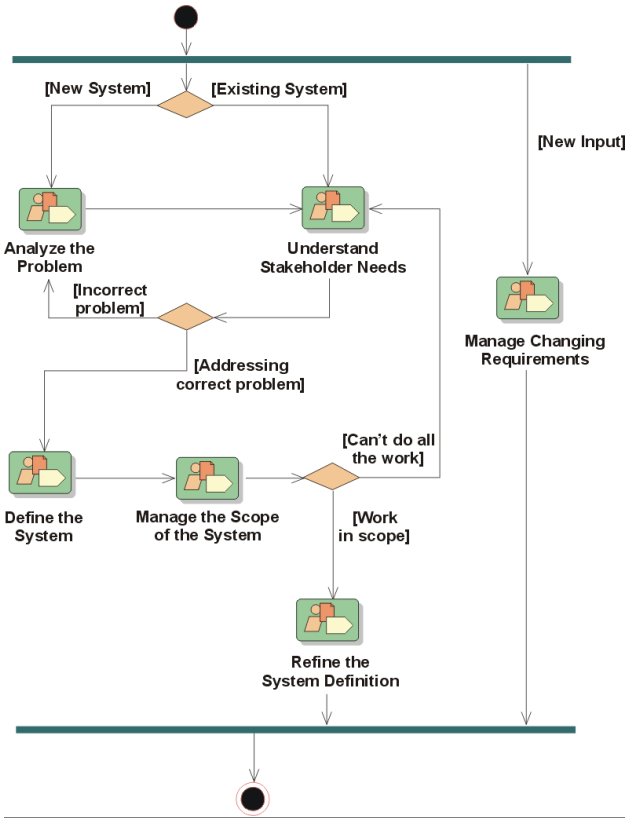

Figura 9—Fluxo de Trabalho de Requisitos no Rational Unified Process

Atividade: Analisar o Problema

Na Análise de Problema, a atividade principal é desenvolver a visão para o projeto. A saída dessa atividade é um documento de visão que identifica a visão de nível alto do usuário ou cliente do sistema a ser construído. A visão expressa os requisitos iniciais como recursos-chave que o sistema deve possuir para resolver os problemas mais críticos e atender às necessidades fundamentais dos envolvidos. O analista de sistemas tem a função principal nessa atividade. O analista de sistemas deve ter conhecimento no domínio do problema e um entendimento do problema e deve poder descrever um processo que acredita que resolverá o problema. O envolvimento ativo de vários envolvidos no projeto é requerido e todos os pedidos relevantes dos envolvidos devem ser considerados.

Para iniciar a tarefa Gerenciar Dependências, os recursos devem receber atributos como base racional, valor relativo ou prioridade e fonte de pedidos. Conforme a visão se desenvolve, o analista identifica usuários e sistemas (os agentes) de possíveis casos de uso. Os agentes são os primeiros elementos do modelo de caso de uso, que definirão os requisitos técnicos funcionais e não funcionais do sistema.

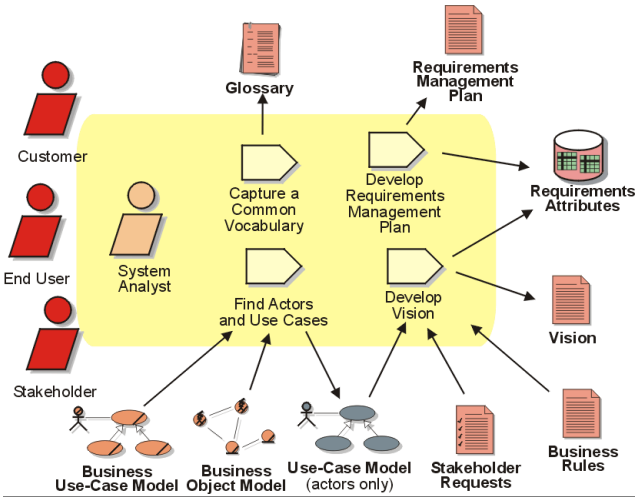

Figura 10—Analisando o Problema

Atividade: Analisar o Problema

Iniciação: Um ou mais envolvidos que percebem um problema iniciarão a atividade.

Um analista de sistemas em uma equipe de desenvolvimento poderá facilitar uma sessão para ajudar os envolvidos iniciais a descreverem o problema que desejam solucionar. É importante haver concordância sobre uma declaração concisa do problema percebido. Elementos-chave de uma **declaração do problema** são mostrados na seguinte tabela:

O Problema de	definir o problema
Afeta os Envolvidos	listar os envolvidos afetados pelo problema
O Impacto do Problema é	descrever o impacto do problema
Uma Solução Bem-sucedida incluirá	listar alguns benefícios fundamentais de uma solução bem-sucedida

A declaração do problema explica concisamente o objetivo do projeto. A análise do problema estimula investigação adicional em todos os pedidos dos envolvidos e no caso de negócios inicial, incluindo benefícios convincentes e custos aproximadamente estimados. Em paralelo com a definição das declarações do problema, a equipe também deve compilar um glossário acompanhando os termos utilizados comumente e concordando com suas definições.

Introdução do Modelo de Caso de Uso

Um modelo de caso de uso consiste em agentes, casos de uso e relacionamentos entre eles. Os agentes representam tudo que deve trocar informações com o sistema, incluindo os que normalmente são chamados de usuários. Quando um agente utiliza o sistema, o sistema executa um caso de uso. Um bom caso de uso é uma seqüência de transações que produz um resultado mensurável do valor para um agente. A coleta de casos de uso é a funcionalidade completa do sistema.

—Jacobson I., Christerson M., Jonsson P., Overgaard G., Object-Oriented Software Engineering – A Use Case Driven Approach, Addison Wesley – ACM Press, 1992

A análise do problema também identifica os agentes principais do sistema. Os agentes são usuários do sistema ou de qualquer outro sistema que trocará informações com ele. Nesse estágio, a análise do problema deve identificar rapidamente algumas formas óbvias que os agentes irão interagir com o sistema. As descrições devem ser orientadas em direção ao processo de negócios em vez de ao comportamento do sistema. Por exemplo, um programa de orçamento pode permitir um tipo de agente para "Criar o orçamento departamental", enquanto outro agente pode "Consolidar os orçamentos departamentais". O analista de sistemas pode posteriormente dividi-los em casos de uso adicionais que se alinham mais significativamente com o comportamento específico do sistema. Por exemplo, "Criar o orçamento departamental" pode resultar em casos de uso do sistema como "Importar informações de planilha" e "Criar visualizações de orçamentos".

A sessão de análise de problemas descrita acima é freqüentemente executada mais de uma vez, talvez com diferentes envolvidos, e entremisturada com sessões da equipe de desenvolvimento interna. O analista de sistemas que conduziu a reunião com os envolvidos liderará uma sessão com membros da equipe de desenvolvimento para visionar uma solução técnica para os problemas, derivar recursos das entradas iniciais dos envolvidos e esboçar a descrição da visão, que é a primeira definição do sistema a ser construído. Para facilitar o entendimento da solução proposta entre os envolvidos iniciais, o analista de sistemas poderá utilizar ferramentas de modelagem ou técnicas de desenho manuais para complementar a descrição da visão.

Os envolvidos iniciais são consultados em vários pontos para ajudar a refinar a descrição do problema e restringir o número e o escopo de possíveis soluções. Os envolvidos e os analistas de sistemas gerenciam dependências nessa atividade negociando a prioridade de recursos-chave e ganhando um entendimento geral dos recursos e esforços necessários para desenvolvê-los. Embora a alteração inevitável das estimativas de prioridade e esforços ou recursos, o gerenciamento de dependências estabelece antecipadamente um padrão importante que continua em todo o ciclo de vida de desenvolvimento. Essa é a essência do gerenciamento de escopo e um preditor antecipado do sucesso de um projeto.

Depois de vários esboços, a visão atinge um ponto no qual a equipe deve decidir se irá investir em levantamento adicional de requisitos. Ao mesmo tempo, o processo de aprovação do caso de negócios foi iniciado separadamente. Embora não discutido adicionalmente neste documento, o caso de negócios descreve o seguinte:

- contexto (domínio, mercado e escopo do produto);
- abordagem técnica;
- abordagem de gerenciamento (planejamento, risco medida objetiva do sucesso);
- previsão financeira.

Atividade: Entender as Necessidades dos Envolvidos

Se a análise do problema inicial justificar gasto adicional, a atividade Entendendo as Necessidades dos Envolvidos é iniciada à sério. A tarefa-chave é fazer o levantamento dos pedidos dos envolvidos. As saídas principais são coletas de necessidades e recursos priorizados dos envolvidos, que permitem o refinamento do documento de visão e fornecem um melhor entendimento dos atributos de requisito. Além disso, durante essa atividade você pode começar discutindo o sistema em termos de casos de uso e agentes. Outra saída importante é um glossário atualizado de termos para facilitar o vocabulário comum entre os membros da equipe.

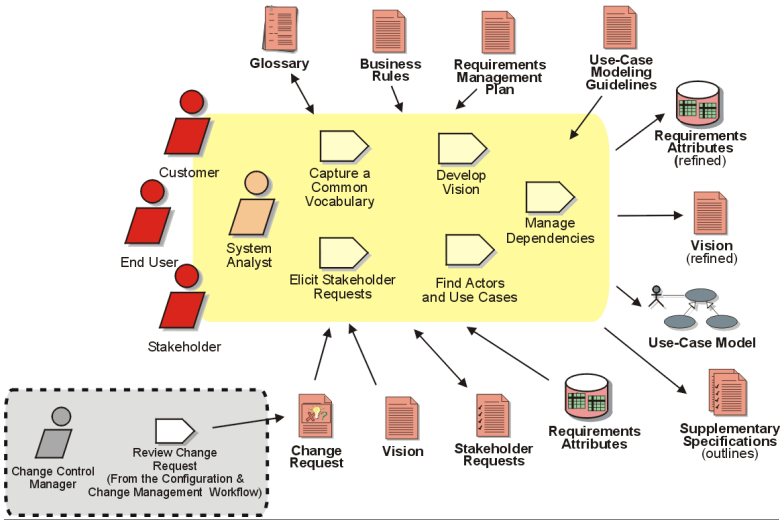

Figura 11—Entendendo as Necessidades dos Envolvidos

Atividade: Entender as Necessidades dos Envolvidos

O analista de sistemas e os envolvidos-chave identificam envolvidos adicionais, fazem o levantamento de seus pedidos e determinam as necessidades e recursos fundamentais através de entrevistas, workshops, storyboards, casos de uso de processos de negócios e outras técnicas. Um ou mais analistas de sistemas auxiliam essas sessões. Esses workshops de requisitos estão entre as técnicas de levantamento mais úteis. O processo inclui usuários, pessoal de helpdesk, proprietários de empresas, testadores e outros que têm um interesse no resultado do projeto proposto. Os pedidos dos envolvidos são freqüentemente ambíguos, sobrepostos e até estranhos. Além dos resultados do levantamento formal, os pedidos dos envolvidos podem ser expressos em documentos bem formatados, em defeitos e pedidos de aprimoramento de bancos de dados ou em encadeamentos de e-mail e groupware. Os analistas de sistemas registram, categorizam e priorizam as necessidades identificadas dos envolvidos-chave.

Entender as Necessidades dos Envolvidos: Onde Começa o "Deleite dos Clientes"

Os pedidos dos envolvidos são capturados ao máximo na linguagem e no formato do envolvido que está fazendo o pedido. Ao contrário dos tipos de requisitos subseqüentes, que normalmente têm autoria dos membros da equipe do projeto treinados para o processo e proficientes tecnicamente, os pedidos dos envolvidos são freqüentemente expressos de forma ruim. Eles são duplicados ou sobrepostos. Eles podem ser expressos sobre qualquer coisa, desde tiras de papel a bancos de dados de pedidos de aprimoramento.

O analista (ou equipe que representa a função de analista) deve revisar todos os pedidos, interpretá-los, agrupá-los, talvez digitá-los novamente (sem reescrevê-los) e traduzi-los para as necessidades e recursos do sistema fundamentais dos envolvidos no documento de visão. Dependendo do rigor aplicado em seu desenvolvimento e na disponibilidade de ferramentas, a rastreabilidade entre alguns pedidos (ou todos) dos envolvidos, as necessidades e os recursos podem ser aplicados para ajudar os envolvidos a entender como seus pedidos e necessidades são considerados na determinação dos requisitos do sistema.

Demonstrar sério interesse para fazer o levantamento de pedidos e satisfazer as necessidades dos envolvidos, aplicando a Atividade Entendendo as Necessidades dos Envolvidos, pode ser crítico para estabelecer a confiança do envolvido nas habilidades da equipe de desenvolvimento.

Baseado em um melhor entendimento das necessidades dos envolvidos, os analistas de sistemas na equipe de desenvolvimento refinam o documento de visão, prestando atenção especial ao desenvolvimento de uma "declaração de posição de produto". Em duas ou três sentenças, essa declaração estabelece o valor convincente do projeto. A declaração deve incluir os usuários planejados, os problemas que ela soluciona, os benefícios que ela fornece e os concorrentes que ela substitui. Todos os membros da equipe devem entender esse tema do projeto. Os analistas de sistemas também atualizam o glossário para facilitar o entendimento comum dos termos.

Os envolvidos-chave são consultados em vários pontos para negociar prioridades de novos recursos derivados do entendimento das necessidades dos envolvidos e ganham um entendimento atual dos recursos e esforços necessários para desenvolvê-los. Como ocorre na análise de problemas, o gerenciamento de dependências nessa atividade ajuda a gerenciar o escopo. Ele também estabelece a rastreabilidade entre os pedidos dos envolvidos, necessidades e recursos do sistema, portanto, os envolvidos podem assegurar-se de que suas entradas foram consideradas.

Atividade: Definir o Sistema

As duas primeiras atividades de Requisitos, Analisar o Problema e Entender as Necessidades dos Envolvidos, criam iterações antecipadas das definições do sistema-chave, incluindo recursos especificados no documento de visão, um primeiro esboço para o modelo de caso de uso e atributos de requisitos iniciais. A próxima atividade, Definir o Sistema, completa a descrição dos requisitos do sistema de nível alto com refinamento das definições de recurso, a adição de novos agentes, casos de uso e especificações suplementares.

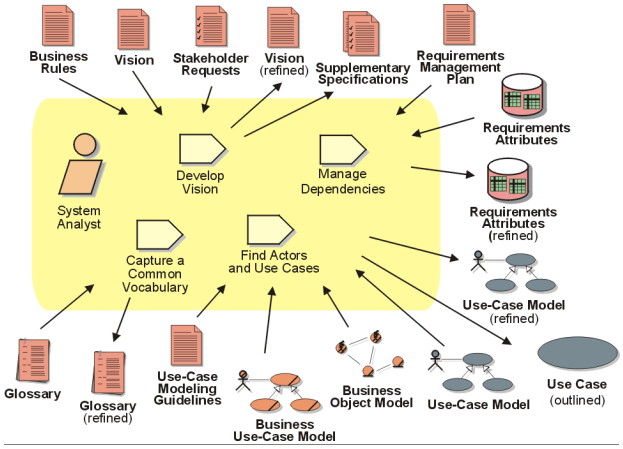

Figura 12—Definindo o Sistema

Atividade: Definir o Sistema

O glossário é atualizado para refletir o entendimento atual sobre os termos utilizados para descrever recursos e os requisitos capturados no modelo de caso de uso e nas Especificações Suplementares.

O analista de sistemas utiliza o conjunto de recursos definido na visão refinada para derivar e descrever os casos de uso que elaboram a visão dos usuários do comportamento esperado do sistema. O modelo de caso de uso serve como um contrato entre o cliente, os usuários e os desenvolvedores do sistema sobre como os recursos selecionados funcionarão no sistema. Ele ajuda a definir expectativas e objetivos realísticos para desenvolvedores do sistema e ajuda os clientes e usuários a confirmarem que o sistema atenderá a essas expectativas.

Alguns requisitos do sistema não se ajustam bem em casos de uso. O analista de sistemas os descreve em uma especificação suplementar. Muitos requisitos não funcionais, como usabilidade, confiabilidade, desempenho e suportabilidade freqüentemente são finalizados aqui. Deve ser observado que muitos requisitos não funcionais desses tipos são específicos a um único caso de uso. É melhor para os especificadores de casos de uso colocar esses requisitos na própria especificação de caso de uso (consulte a atividade Refinando o Sistema), deixando a especificação suplementar para requisitos não funcionais em todo o sistema.

Nessa atividade, o analista de sistemas cria e define atributos para requisitos suplementares (como prioridade e casos de uso relacionados). Além disso, o analista de sistemas inclui e atualiza valores de atributos para os casos de uso iniciais e novos.

Finalmente, o analista de sistemas gerencia dependências rastreando necessidades importantes do usuário e recursos críticos para casos de uso relacionados e requisitos descritos em especificações suplementares.

Atividade: Gerenciar o Escopo do Sistema

Depois de identificar os requisitos no nível do recurso, descrever a maioria dos agentes, casos de uso e outros requisitos especificados nas especificações suplementares, o analista de sistemas deve reunir e designar valores o mais precisamente possível aos atributos de requisito, como prioridade, esforço, custo e risco. Isso permite um melhor entendimento sobre como determinar o escopo inicial do release do sistema e também pode permitir que o arquiteto identifique casos de uso arquiteturalmente significativos.

O plano de iteração, desenvolvido em paralelo pelo gerenciamento de projeto e de desenvolvimento, aparece primeiro nessa atividade: Gerenciar o Escopo do Sistema. Também conhecido como um plano de desenvolvimento, o plano de iteração define o número e a freqüência de iterações planejadas para o release. Os elementos de maior risco dentro do escopo devem ser planejados para iterações antecipadas.

Outras saídas importantes da atividade Gerenciando o Escopo incluem a iteração inicial do documento da arquitetura de software e uma visão revisada que reflete o entendimento crescente dos analistas e envolvidos-chave sobre a funcionalidade do sistema e os recursos do projeto.

Como ocorre no caso de negócios, mencionado anteriormente, e no primeiro problema do plano de iteração, o documento da arquitetura de software não é um artefato do gerenciamento de requisitos, embora esteja relacionado e faça parte do Rational Unified Process. Ele não é assunto desse documento.

> A experiência ensina que as chaves para gerenciar o escopo com êxito são valores de atributo bem considerados designados às necessidades dos envolvidos, recursos, casos de uso e outros requisitos especificados em especificações suplementares; juntamente com iteração regular, aberta e honesta com os envolvidos representativos.

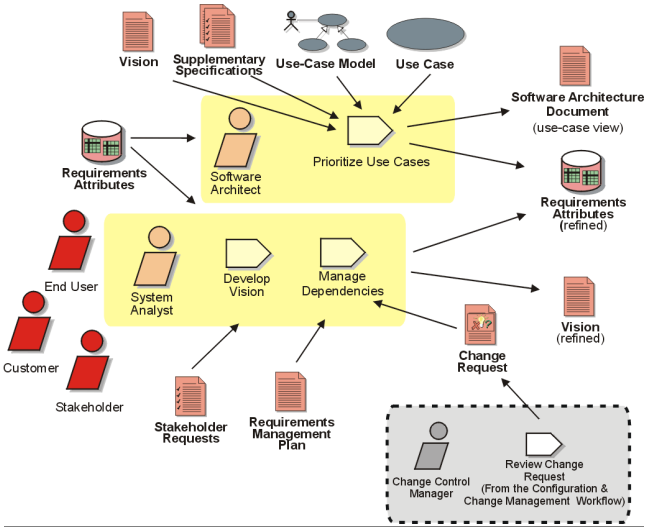

Figura 13—Gerenciando o Escopo do Sistema

Atividade: Gerenciar o Escopo do Sistema

Os arquitetos priorizam casos de uso para sua cobertura de risco, significância arquitetural e cobertura arquitetural. Enquanto o sistema pode ser definido com vários requisitos de casos de uso e especificações suplementares, apenas um subconjunto de casos de uso é normalmente crítico para uma boa arquitetura de sistema. Com casos de uso priorizados, os arquitetos refinam a iteração ou o plano de desenvolvimento e modelam uma visualização de caso de uso da arquitetura do sistema em ferramentas, como o Rational Rose.

Os analistas de sistemas gerenciam dependências refinando atributos de requisitos para recursos na visão. Eles também refinam requisitos em casos de uso e em especificações suplementares. Os analistas de sistemas asseguram que existe rastreabilidade* adequada para os requisitos de necessidades, recursos e casos de uso dos envolvidos e requisitos de especificações suplementares. O termo "rastreabilidade adequada" é deliberado. Consulte o texto em Rastreabilidade posteriormente neste documento.

Esta etapa está entre a mais importante no projeto inteiro. Pela primeira vez, a amplitude de conhecimento sobre o sistema proposto está disponível para tornar sérios os compromissos de requisitos, recursos do projeto e datas de entrega. Ao mesmo tempo, deve-se entender que esses requisitos serão alterados conforme aumenta a profundidade de conhecimento. Se as dependências tiverem sido gerenciadas nas três atividades anteriores, esta etapa será bem mais fácil e as alterações futuras serão mais fáceis.

Durante todo o projeto, conforme são alteradas as situações e ambientes, essa atividade é revisitada várias vezes porque o analista de sistemas deve negociar o escopo e a visão do projeto revisado com os envolvidos-chave. O gerenciamento do escopo do projeto para que corresponda aos recursos disponíveis será bem-sucedido apenas se os envolvidos e os membros da equipe de desenvolvimento visualizem essa etapa como uma progressão natural – não como uma cilada para as expectativas dos usuários ou uma tentativa de extorquir da organização mais tempo e dinheiro. Essa atividade precisará ser repetida em marcos principais no projeto para avaliar se a nova percepção do sistema e de seus problemas requer alteração no escopo. Como requisitos, orçamentos e prazos finais confirmados são difíceis de serem alterados, um entendimento profundo de casos de uso priorizados, de especificações suplementares e de iterações antecipadas do sistema inevitavelmente leva à reconsideração do escopo.

Novamente, é importante que a equipe do projeto se comprometa com o gerenciamento habitual do escopo antes de chegar ao estágio de refinamento, bem como com todo o ciclo de vida do projeto conforme ocorrem as alterações. Os envolvidos representativos devem entender e confiar que suas prioridades e interesses são considerados seriamente durante as negociações de escopo cada vez mais difíceis. Enquanto os requisitos do sistema são refinados, apenas requisitos importantes permanecem para serem negociados ou modificados. A menos que hábitos efetivos de gerenciamento de escopo tenham sido estabelecidos, seu projeto poderá estar destinado a uma "marcha da morte" – um projeto irremediavelmente com excesso de escopos movendo-se inexoravelmente em direção a atrasos e excessos de custo.

Atividade: Refinar a Definição do Sistema

Seguindo em direção a Refinar a Definição do Sistema, os detalhes dessa atividade assumem que os casos de uso no nível do sistema foram esboçados e os agentes foram descritos, ao menos brevemente. Através do gerenciamento do escopo do projeto, os recursos na visão foram novamente priorizados e agora acredita-se que sejam realizáveis com orçamentos e datas completamente seguros. A saída dessa atividade é um entendimento mais profundo da funcionalidade do sistema expressa em casos de uso detalhados, especificações suplementares revisadas e iterações antecipadas do próprio sistema.

Obviamente, nem todos os sistemas terão interfaces com o usuário e nem todas as iterações antecipadas incluirão elementos da GUI. Nós os utilizamos aqui apenas como um exemplo de uma iteração antecipada. Outros exemplos incluem protótipos, modelos e storyboards.

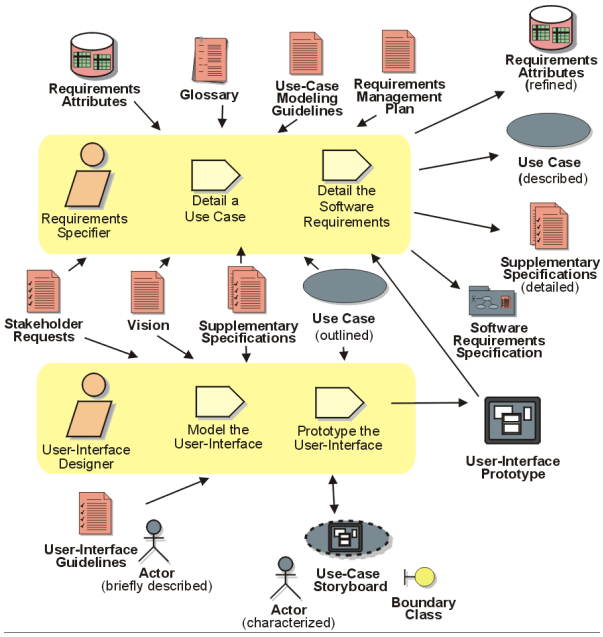

Figura 14—Refinando a Definição do Sistema

Atividade: Refinar a Definição do Sistema

O especificador de caso de uso detalha a definição do fluxo de eventos, pré e pós-condições e outras propriedades textuais de cada caso de uso. Para minimizar o esforço e aprimorar a capacidade de leitura, é aconselhável utilizar um formato de documento padrão ou um gabarito de especificação de caso de uso para capturar informações textuais sobre cada caso de uso. A criação de especificações de caso de uso bem consideradas é crítica para a qualidade do sistema. Esse desenvolvimento de especificação requer um entendimento completo das necessidades e dos recursos dos envolvidos relacionados ao caso de uso. É desejável que vários membros da equipe do projeto (como engenheiros de software) participem na criação de casos de uso.

Em paralelo, o especificador de caso de uso revisa a especificação suplementar com requisitos adicionais que não são específicos para o caso de uso.

O designer da interface com o usuário modela e utiliza um protótipo da interface com o usuário do sistema. Esse trabalho está altamente correlacionado com a evolução dos casos de uso.

O especificador de caso de uso e o analista de sistemas revisam o esforço, o custo, o risco e outros valores de atributo para cada requisito que é entendido melhor.

Os resultados desse refinamento da definição do sistema são enviados para outra rodada da atividade Gerenciando o Escopo. Depois de saber mais sobre o sistema, você poderá desejar alterar as prioridades. Mais definitivamente, o escopo do release do sistema precisará ser revisado e refinado, se necessário.

Atividade: Gerenciar a Alteração de Requisitos

Conforme ocorrem as alterações—e elas ocorrerão inevitavelmente—a atividade Gerenciar a Alteração de Requisitos precisa ser aplicada continuamente durante toda a vida do projeto, conforme discutido em Gerenciar o Escopo do Sistema. A saída dessa atividade pode causar modificações em cada artefato, que, por sua vez, requer comunicação efetiva entre todos os membros da equipe do projeto e envolvidos.

Nessa atividade, introduzimos artefatos adicionais que são afetados pelas atividades de requisitos. As alterações em requisitos afetam naturalmente os modelos de sistema que são representados nas atividades de análise & design. As alterações de

requisito também afetam os testes criados para validar a implementação adequada dos requisitos. Como nos exemplos anteriores, esses artefatos são parte do Rational Unified Process, mas não são os assuntos deste documento. Os relacionamentos de rastreabilidade identificados no processo de gerenciamento de dependências são as chaves para entender esses impactos.

Rastreabilidade

Muito é feito da rastreabilidade no campo de requisitos. Muitos promovem a eficácia do rastreamento de requisitos individuais do cliente para cada especificação, teste, elemento de modelo e, essencialmente, arquivos de código fonte relacionados. Certamente, alguma rastreabilidade é a chave para o gerenciamento bem-sucedido de alterações de requisitos.

Saiba, no entanto, que todas as formas de rastreabilidade requerem um gasto com configuração e manutenção durante a vida do projeto. Como em todos os gastos, a rastreabilidade diminui pontos de retorno dependendo de sua situação específica. Este documento enfatiza o valor do rastreamento entre tipos diferentes de requisitos. Esse é um bom local para começar e pode ser automatizado por ferramentas, como o Rational RequisitePro, o Rational Rose, o Rational SoDA e o Rational TeamTest. Acreditamos que você achará que algum nível de rastreabilidade de requisitos é um bom gasto.

Para obter informações adicionais sobre estratégias diferentes para rastreabilidade de requisitos, consulte o white paper, "Estratégias de Rastreabilidade para Gerenciar Requisitos com Casos de Uso" [6].

Field Code Changed

Outro conceito importante para Gerenciar a Alteração de Requisitos é o rastreamento do histórico de requisitos. Pela assimilação da natureza e base racional das alterações de requisitos, os revisores (alguém da equipe de projeto de software cujo trabalho seja afetado pela alteração) recebem as informações necessárias para responder à alteração corretamente.

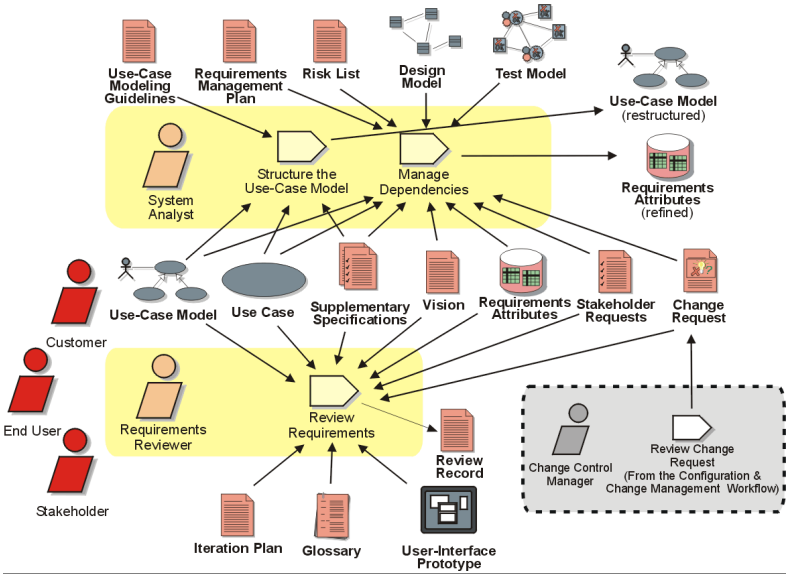

Figura 15—Gerenciando Alteração de Requisitos

Atividade: Gerenciar a Alteração de Requisitos

Os pedidos de alterações em requisitos podem ser iniciados por qualquer envolvido ou membro da equipe do projeto por diversas razões. Todos os CRs (Controle de Alterações), incluindo alterações em requisitos ou em pedidos de aprimoramento, bem como defeitos, devem passar pelo mesmo processo CRM (Gerenciamento de Controle de Alterações). No mínimo, ele deve incluir o registro e o rastreio do pedido em um sistema de banco de dados centralizado e deve forçar a revisão por uma autoridade de revisão central. Consulte outras seções do Rational Unified Process para obter detalhes adicionais de tal processo CRM.

O analista de sistemas deve coordenar a atividade de revisão, preferivelmente por um CCB (Conselho de Controle de Alterações), coletando e revisando todos os controles de alterações, e classificá-los como:

- defeitos na implementação que não afetam os requisitos
- modificações de algum tipo em requisitos existentes
- solicitações de melhoria

Depois de classificadas, as alterações de requisitos propostas recebem atributos e valores, conforme descrito em outras atividades de requisitos.

Na revisão dos controles de alterações, o analista de sistemas apresenta as alterações de requisitos priorizadas propostas a um CCB composto de envolvidos representativos e membros da equipe do projeto. As modificações de escopo que excedem os recursos devem ser rejeitadas ou levantadas para os representantes dos envolvidos que podem aprovar as alterações requeridas para compromissos de datas e orçamentos.

O CCB aprova ou rejeita as alterações de requisitos.

O analista de sistemas comunica as alterações de requisitos ao especificadores de requisitos ou faz as alterações diretamente nos requisitos na visão, nos casos de uso, nos documentos de especificação suplementar ou em outros artefatos de requisitos.

Os revisores de requisitos (desenvolvedores, testadores, gerentes e outros membros da equipe do projeto) avaliam o impacto das alterações dos requisitos em seu trabalho, revisando o histórico de requisitos. implementarFinally, eles implementam a alteração e fazem as alterações necessárias aos requisitos relativos para os quais tem autoridade.

Sumário

A necessidade de gerenciar requisitos não é nova. Portanto, o que faz com que as informações precedentes sejam consideradas agora?

Primeiro, se seus projetos não estiverem satisfazendo os clientes regularmente, atendendo aos prazos finais e ficando dentro do orçamento, você tem razões para reconsiderar a sua abordagem de desenvolvimento. Se ao fazer isso, você determinar que os problemas relacionados aos requisitos estão minando os seus esforços de desenvolvimento, você tem razões para considerar boas práticas de gerenciamento de requisitos.

Segundo, as práticas de gerenciamento de requisitos resumidas neste documento englobam a experiência coletiva de milhares e são as opiniões bem consideradas de uma série de indivíduos que passaram anos trabalhando com clientes no campo de gerenciamento de requisitos. Sugerimos que essa visão geral de suas contribuições—e a mais direta apresentação delas feita no Rational Unified Process—representem uma "boa prática" no gerenciamento de requisitos. Você não encontrará mais pareceres comprovados sobre requisitos em nenhum lugar.

Os autores gostariam de agradecer as contribuições diretas e indiretas do Dr. Ivar Jacobson e de Dean Leffingwell, Dr. Alan Davis, Ed Yourdon e Elemer Magaziner. O mais importante é que apreciamos os clientes da Rational Software que aplicaram e aprimoraram essas práticas em centenas de projetos de desenvolvimento.

Finalmente, nos dois últimos anos, a Rational Software, uma líder na produção de soluções efetivas de desenvolvimento de software, assumiu o desafio de gerenciamento de requisitos e surgiu com as ferramentas para automatizar essa tarefa difícil. Os problemas crônicos e difusos do gerenciamento de requisitos são solucionáveis. E, essencialmente, pode ser a melhor razão para iniciar a prática de excelência em gerenciamento de requisitos nos dias de hoje.

Para obter uma discussão completa dos conceitos acima, consulte o excelente livro de Dean Leffingwell sobre Managing Software Requirements [7].

Field Code Changed

Referências

[1] CHAOS, The Standish Group International, Inc., Dennis, MA, 1994, 1997.

[2] Computer Industry Daily, 12 de dezembro de 1997.

[3] Dorfman, M. e R. Thayer, *Software Engineering*, IEEE Computer Society Press, Los Alamitos, CA, 1997 pp. 79.

[4] Dorfman, M. e R. Thayer, *Software Engineering*, IEEE Computer Society Press, Los Alamitos, CA, 1997 pp. 80.

[5] Spence, Ian e Leslee Probasco, *Estratégias de Rastreabilidade para Gerenciar Requisitos com Casos de Uso*, White Paper, Rational Software Corporation, 1998.

[6] Rational Unified Process®, Rational Software Corporation, Cupertino, CA, 1999.

[7] Leffingwell, Dean e Don Widrig, *Managing Software Requirements — A Unified Approach*, Addison-Wesley, 2000.

Rational®
the software development company

Duas Sedes:

Rational Software
18880 Homestead Road
Cupertino, CA 95014
Tel: (408) 863-9900

Rational Software
20 Maguire Road
Lexington, MA 02421
Tel: (781) 676-2400

Sem custo: (800) 728-1212
E-mail: info@rational.com
Web: www.rational.com
Localização Internacional: www.rational.com/worldwide

Field Code Changed

Field Code Changed

Field Code Changed