Desenvolvendo Sistemas em Larga Escala com o Rational Unified Process

Maria Ericsson

Rational Software White Paper

TP 156

Rational®
the software development company

Índice Analítico

Histórico

Este documento foi feito no "Systems of Interconnected Systems", publicado no ROAD, Maio-Junho de 1995, de Ivar Jacobson, Karin Palmkvist e Susanne Dyrhage [1]. Este documento é beneficiado pela entrada de vários projetos de desenvolvimento de sistemas em larga escala e também tem por objetivo alinhar o Rational Unified Process versão 5.1 [2] e a Unified Modeling Language [3].

Sistemas de Sistemas Interconectados

Há um aumento considerável de complexidade no desenvolvimento de sistema de larga escala. Não só é necessário que você seja capaz de compreender um conjunto mais complexo de artefatos, mas que também introduza uma sobrecarga, pois precisa gerenciar um conjunto maior de recursos. Esse artigo descreve um padrão de arquitetura que é usado para ajudar a controlar a sobrecarga adicional de complexidade. O padrão arquitetônico, entre outros locais discutidos no [4], é utilizado como referência como um **sistema de sistemas interconectados**.

Essa construção é útil ao construir sistemas muito grandes ou complexos como os sistemas de comando e de controle ou soluções de TI altamente integradas. Esse tipo de "super sistemas" é na maioria dos casos dividido em várias partes separadas, cada um desenvolvido independentemente como um sistema separado. Um super sistema é implementado por um conjunto de sistemas interconectados, comunicando uns com os outros para preencher as obrigações do super sistema. Um desses sistemas representa os recursos gerais e o chamamos de **sistema subordinador**. Os outros sistemas representam uma parte do todo e os chamamos de **sistemas subordinados**. Um sistema subordinador é claramente distinto dos sistemas subordinados que o implementam. A relação entre os diferentes tipos de sistemas é distinta: a partir da perspectiva dos sistemas subordinadores, os sistemas subordinados são subsistemas, consulte a figura 1.

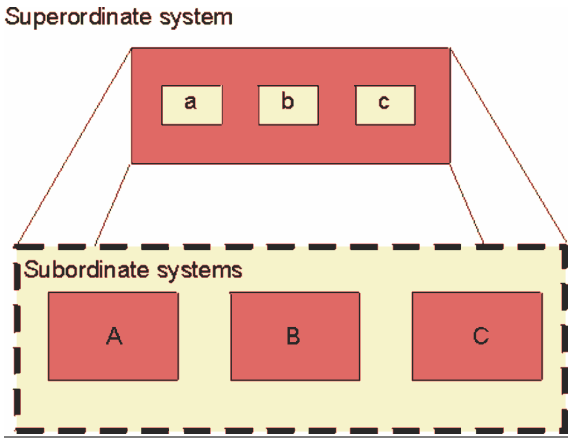

Figura 1. A especificação de um sistema subordinador é implementada por um sistema de sistemas interconectados em que os sistemas A, B, e C sejam implementações dos subsistemas a, b e c do sistema subordinador respectivamente.

Separar o sistema subordinador de seus sistemas subordinados tem várias vantagens:

- Os sistemas subordinados podem ser gerenciados separadamente durante todas as atividades do ciclo de vida, incluindo vendas e entrega.
- Facilita a utilização de um sistema subordinado para implementar outros sistemas subordinadores, ligando-o a outros sistemas de sistemas interconectados.

- Nem sempre você sabe quando começar a construir um sistema, se é um sistema de sistemas interconectados ou não. É possível começar a trabalhar com uma visualização do sistema "simples" e posteriormente, no ciclo de vida, determinar se é preciso aplicar o padrão do sistema de sistemas interconectados ou não.
- Permite que você faça mudanças internas nos sistemas subordinados sem desenvolver uma nova versão do sistema subordinador. As novas versões dos sistemas subordinadores são necessárias somente por causa de mudanças funcionais maiores.

Cada sistema subordinado tem um conjunto associado de artefatos, com clara rastreabilidade entre eles. Há também a rastreabilidade mantida a partir dos conjuntos de artefato dos sistemas subordinados para os conjuntos de artefato correspondentes do sistema subordinador. Cada sistema subordinado pode ser gerenciado como um projeto de desenvolvimento separado com suas próprias fases do ciclo de vida: iniciação, elaboração, construção e transição.

Se o "super sistema" que você está construindo for muito grande, um sistema subordinado pode precisar ser dividido e ser tratado como um sistema de sistemas interconectados.

O Ciclo de Vida de Desenvolvimento de Software

No Rational Unified Process, o ciclo de vida de desenvolvimento é apresentado e discutido a partir de duas perspectivas: a perspectiva de gerenciamento e a perspectiva de desenvolvimento, consulte a figura 2.

A partir de uma perspectiva de gerenciamento, passe por quatro fases do ciclo de vida para desenvolver um sistema ou uma nova geração de um sistema. A partir da perspectiva de desenvolvimento, desenvolva versões do sistema iterativamente que são incrementalmente mais e mais completas. As atividades que você realizar durante uma iteração são agrupadas no Rational Unified Process em um conjunto de disciplinas. Cada disciplina focaliza a descrição de alguns aspectos do sistema, resultando em um modelo do sistema ou um conjunto de documentos.

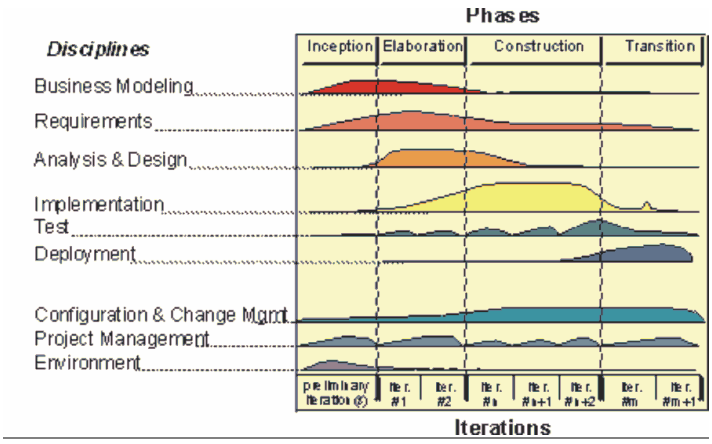

Figura 2. O modelo iterativo

Aplicar isso aos sistemas de sistema interconectado, ao sistema subordinador assim como cada um de seus sistemas subordinados, passa por todo o ciclo de vida e é freqüentemente tratado como projetos separados.

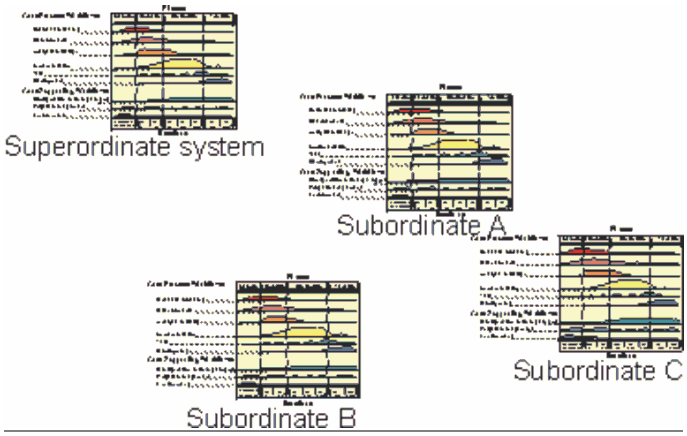

Figura 3. Cada sistema, subordinador e subordinado, passa pelo próprio ciclo de vida.

Os ciclos de vida têm dependências, é claro. Gerenciar essas dependências corretamente é um dos desafios de desenvolver um sistema de sistemas interconectados. As dependências são do seguinte tipo:

- Os ciclos de vida são dependentes em tempo. O ciclo de vida do sistema subordinador começa primeiro. Assim que o sistema subordinador tiver passado por pelo menos uma iteração e as interfaces dos sistemas subordinados estiverem relativamente estáveis, os ciclos de vida dos sistemas subordinados podem começar. Na verdade, você pode nem saber quais são os sistemas subordinados até ter passado por pelo menos uma iteração no sistema subordinador.
- O ciclo de vida do sistema subordinador pode entrar em manutenção uma vez que as interfaces dos sistemas subordinados estejam estáveis. Isso significa que nenhum desenvolvimento ativo é feito, somente se problemas que requerem mudanças das interfaces de sistemas subordinados ocorrerem.
- As interfaces dos sistemas subordinados são de propriedade das pessoas que desenvolvem o sistema subordinador. Para saber mais sobre interfaces, consulte [3] e [5].
- As classes que implementam as interfaces do sistema subordinado são de propriedade das pessoas que desenvolvem os sistemas subordinados.

Workflows e Artefatos de Desenvolvimento do Sistema

Uma suposição natural é que o sistema subordinador, assim como os sistemas subordinados, pode ser desenvolvido com o mesmo conjunto de artefatos e desenvolvido pelos mesmos workflows que são típicos para sistemas não compostos. Antes que possamos prosseguir mostrando como isso pode ser feito, esses artefatos e workflows têm que ser introduzidos. No Rational Unified Process, introduzimos cinco disciplinas, veja a figura 4. São elas:

- Modelagem de negócio – a finalidade é avaliar a organização em que o sistema será utilizado, para compreender melhor as necessidades e os problemas que serão resolvidos pelo sistema. O resultado é um modelo de caso de uso de negócio e um modelo de análise de negócio. Essa disciplina pode ser considerada opcional. Se a organização em que o sistema será empregado for muito simplista, pode não adicionar valor.
- Requisitos – com a finalidade de capturar e avaliar os requisitos, colocando a usabilidade em foco. Isso resulta em um modelo de caso de uso com atores representando unidades externas se comunicando com o sistema e os casos de uso representando seqüências de transação, rendendo resultados mensuráveis de valor para os atores.

- Análise e Design –com a finalidade de investigar o ambiente de implementação pretendido e o efeito que ele terá na construção do sistema. Isso resulta em um modelo de objeto (o modelo de design), incluindo as realizações de caso de uso que mostram como os objetos se comunicam para executar o fluxo dos casos de uso. Isso pode incluir definições de interface para classes e subsistemas, especificando suas responsabilidades em termos de operações fornecidas. Esse modelo de objeto também é adaptado para o ambiente de implementação em termos de linguagem de implementação, distribuição, etc. Às vezes é útil considerar os resultados das análises de um modelo separado e, em seguida, chamar o modelo de análise.

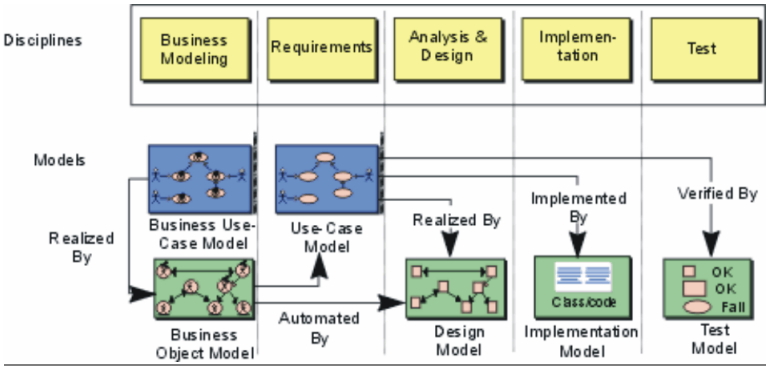

Figura 4. Cada disciplina está associada a um conjunto específico de modelos.

- Implementação – com a finalidade de implementar o sistema no ambiente de implementação prescrito. Isso resulta em código fonte, executáveis e arquivos.
- Teste –com a finalidade de assegurar que o sistema é o pretendido e que não há erros na implementação. Isso resulta em um sistema certificado que está pronto para entrega.

Desenvolvimento de um Sistema de Sistemas Interconectados.

O que realmente devemos fazer é definir como as responsabilidades do sistema podem ser distribuídas em vários sistemas, cada um tomando conta de um subconjunto bem definido dessas responsabilidades. Isso significa que o objetivo principal é definir as interfaces entre esses sistemas subordinados. Quando isso é concluído, o restante do trabalho pode ser feito separadamente para cada sistema subordinado, de acordo com o princípio de "dividir e conquistar". Portanto, isso é tudo que devemos fazer para o sistema como um todo, além de testá-lo assim que a implementação é feita.

Critérios para Decomposição

Então como decidir se o sistema deve ser decomposto em um sistema de sistemas interconectados? Há algumas características que devem ser consideradas:

- Para um sistema de tamanho e complexidade consideráveis, é possível particionar o problema em pedaços menores que são mais fáceis de compreender um de cada vez.
- Você está lidando com sistemas separados fisicamente? Freqüentemente esse é o caso ao trabalhar com sistemas legados ou arquitetura legada.
- Decompor ajuda a definir interfaces naturais e estreitas entre partes do sistema.
- Você pode decidir implementar uma parte do sistema utilizando alguns produtos COTS (Commercial-Off-The-Shelf) maiores. Decompor ajudará a esclarecer como você pretende utilizar o produto COTS.

- Particionar permite que você obtenha a maior parte de uma organização de desenvolvimento distribuído e particione claramente o trabalho entre várias equipes dispersas geograficamente.

Os riscos a serem considerados são:

- A utilização excessiva da decomposição pode ocultar o problema geral para todos os detalhes.
- Utilizando sistemas separados fisicamente ou equipes separadas fisicamente, você corre o risco de eliminar qualquer forma de reutilização e acabar com um sistema de tubo de aquecimento rígido.

Organização

Para mitigar os riscos mencionados anteriormente, é essencial ter um grupo de pessoas designados para supervisionar todo o esforço de desenvolvimento. Este grupo é freqüentemente chamado de uma equipe de arquitetura e deve enfatizar os seguintes interesses principais:

- Há uma arquitetura geral definida que é seguida nos sistemas subordinados.
- Há uma ênfase razoável na reutilização e compartilhamento da experiência entre os sistemas subordinados.
- Há um entendimento claro do que os artefatos produzem e quais relações estão entre os artefatos dos sistemas subordinados e subordinadores.
- Uma estratégia de gerenciamento de mudanças efetiva é definida e seguida por todas as equipes.

A equipe de arquitetura pode, mas não sempre, ter o desenvolvimento do sistema subordinador. Para uma discussão mais completa sobre a organização, consulte [6].

O Ciclo de Vida do Sistema Subordinador

Primeiro, você pode opcionalmente fazer a **modelagem de negócio** para entender melhor o contexto do sistema. Isso adiciona valor se:

- houver uma necessidade que os desenvolvedores entendam melhor a organização,
- a organização em si é heterogênea em como faz seus negócios e a terminologia e os processos precisam ser alinhados ou
- o esforço de engenharia de software é feito em conjunto com um esforço de reengenharia de negócios.

Consulte também [6].

Esse esforço resultaria em um modelo de caso de uso de negócio e um modelo de análise de negócio. Alternativamente, você pode optar por fazer uma engenharia de negócios limitada, verificando somente os conceitos principais no domínio de negócios e documentar isso em um modelo de análise de negócio. Isso é freqüentemente conhecido como modelagem de domínio.

Assim que você tiver "acionado o mecanismo" com um conjunto de modelos de negócios, você precisará começar a produzir **requisitos** em todo o sistema. Temos a mesma necessidade de modelagem de requisitos para um sistema de sistemas interconectados que para qualquer outro sistema. Um modelo de caso de uso é uma maneira muito natural de expressar os resultados, consulte [7]. A maneira mais direta de verificar esse modelo de caso de uso subordinador é supor que ele capture completamente os requisitos comportamentais do sistema. No entanto, provavelmente esse raramente é o caso. Já que precisamos implementar o sistema com outros sistemas, o sistema geral é provavelmente bem complexo. Portanto, não é uma boa idéia tentar ser exaustivo neste nível. Assim, um modelo de caso de uso subordinador normalmente fornece um panorama completo, mas simplificado, dos requisitos funcionais do sistema. Não há necessidade de ser muito detalhado neste nível, já que a modelagem detalhada será executada dentro de cada um dos sistemas subordinados implementados. Também é freqüentemente verdadeiro que muitos requisitos não são necessariamente visíveis em um caso de uso subordinador que corta vários subsistemas. Tais requisitos podem ser chamados de "locais" em um subsistema.

A finalidade da**análise e design** é alcançar uma arquitetura robusta do sistema que é, é claro, de importância vital para um sistema de sistemas interconectados. Os desenvolvedores do sistema subordinador devem alcançar uma estrutura robusta de sistemas subordinados enquanto não precisam se preocupar com as estruturas internas. Portanto, modelaremos uma divisão do sistema em partes menores utilizando subsistemas. Para obter o conjunto correto de subsistemas e obter uma primeira idéia de

como distribuir as responsabilidades do sistema subordinador nesses subsistemas, desenvolvemos um modelo de análise. As classes de análise devem representar as funções exercidas por coisas no sistema quando os casos de uso de alto nível são executados. Portanto, o modelo de análise fornece um panorama simplificado da estrutura completa do objeto em analogia ao modelo de caso de uso de alto nível.

As classes de análise relacionadas funcionalmente são agrupadas nos subsistemas. Assim obtemos uma estrutura de subsistema que é ideal no sentido de que possui como base somente critérios funcionais, por exemplo, não levamos em consideração quaisquer requisitos de distribuição. Um fato que é freqüentemente, altamente influente é a existência de sistemas legados. Os sistemas legados podem preencher algumas ou muitas responsabilidades definidas no modelo de análise. A existência desses sistemas pode até mesmo causar um reparticionamento das responsabilidades encontradas nas análises para que você possa alcançar a reutilização máxima dos recursos existentes.

O resultado do design pode ser uma estrutura do subsistema que esteja muito diferente da que definimos com base em critérios funcionais durante a análise. Assim, acabamos com uma estrutura de subsistemas de design que serão implementados por um sistema subordinado, consulte a figura 5. Para conseguir continuar o trabalho de desenvolvimento para cada sistema separadamente, as interfaces são definidas para cada subsistema. Na verdade, a definição das interfaces é a atividade mais importante executada no nível subordinador, já que as interfaces fornecem regras para o desenvolvimento dos sistemas subordinados. Nenhuma classe de design é definida, a única coisa que você faz é definir as interfaces dos subsistemas de design.

Nenhuma **implementação** é feita como parte do ciclo de vida do sistema subordinador, além de, talvez, algum trabalho de protótipo para explorar aspectos técnicos específicos do sistema.

A disciplina final é o **teste**que, neste caso, significa o teste de integração quando os sistemas subordinados diferentes são montados e também o teste que todo o caso de uso subordinador executa de acordo com suas especificações pelos sistemas interconectados em cooperação.

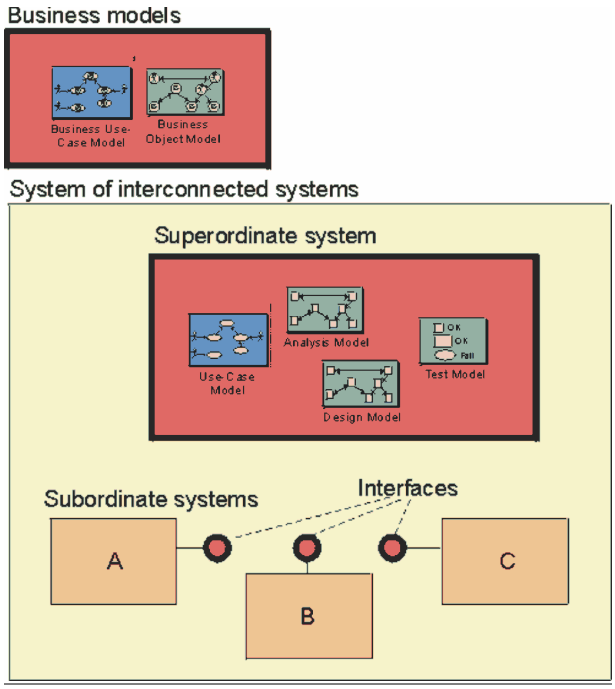

Figura 5. O sistema subordinador é descrito por um conjunto de modelos em que os subsistemas definidos no modelo de design de alto nível serão implementados por sistemas subordinados. As interfaces para os sistemas subordinados são de propriedade do sistema subordinador.

Para mostrar como você trabalharia com o sistema subordinador, aqui estão vários planos de iteração de amostra; um para uma iteração na fase de iniciação do ciclo de vida do sistema subordinador, uma para uma iteração na fase de elaboração. Utilizamos diagramas de atividade para descrever os planos de iteração. Os estados de ação nesses diagramas correspondem às atividades, conforme definido no Rational Unified Process.

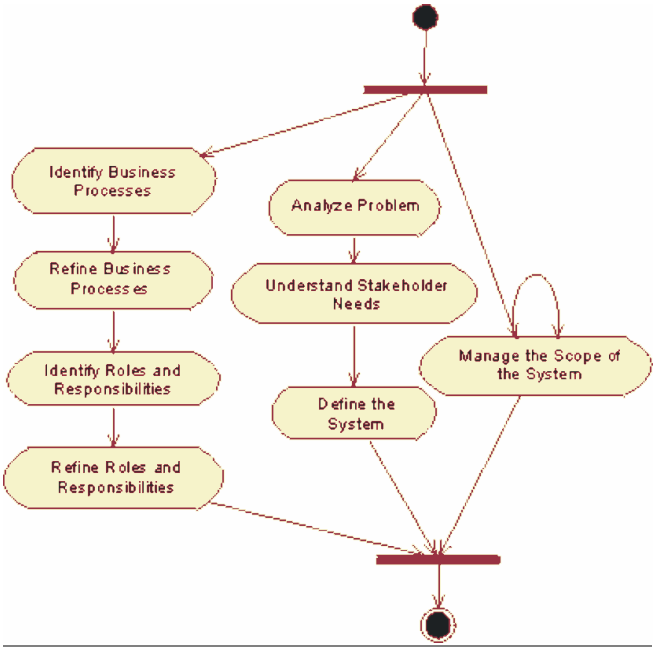

Figura 6. Um diagrama de atividade descrevendo um exemplo de um plano de iteração de iniciação para o sistema subordinador.

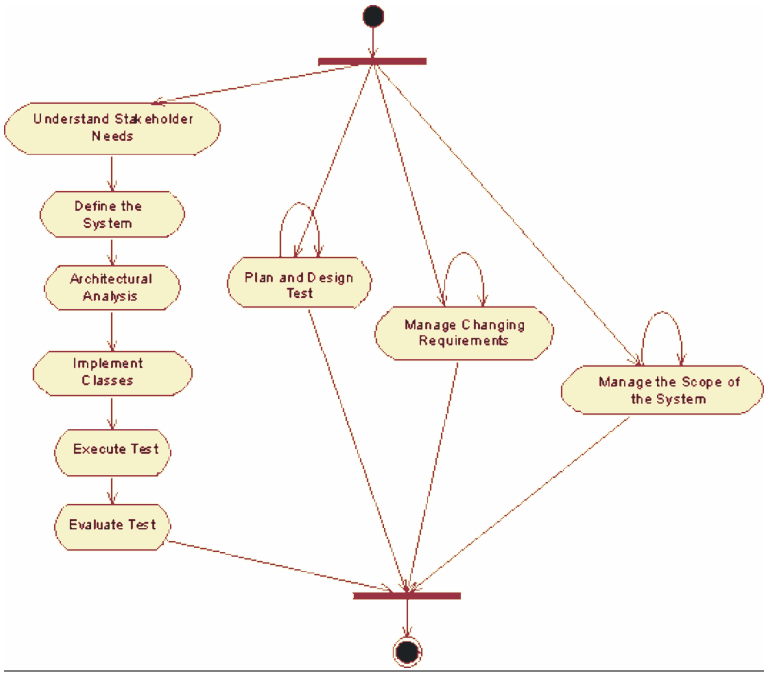

Figura 7. Um diagrama de atividade descrevendo um exemplo de um plano de iteração de elaboração para o sistema subordinador. O estado da ação de "implementar classes" está aqui desde que você possa fazer alguma implementação limitada de protótipos para explorar aspectos técnicos do sistema.

O Ciclo de Vida do Sistema Subordinado

Cada sistema subordinado é desenvolvido da maneira comum, como uma caixa preta, considerando outros sistemas com os quais ele se comunica como ator. Execute o conjunto comum de atividades e desenvolva o conjunto comum de modelos, conforme descrito anteriormente, para cada sistema. Se os modelos no nível subordinador estiverem definidos em todos os detalhes, você obterá recursão completa entre os modelos em níveis diferentes, mas conforme mencionado anteriormente, na prática, esse é raramente o caso.

Para o sistema subordinado, você executará as atividades relacionadas aos **requisitos** . As interfaces e os casos de uso do sistema subordinador serão sua entrada principal para entender os limites do sistema subordinado e quais são seus atores.

Ao executar a **análises & design** para o sistema subordinado, as interfaces definidas no sistema subordinador serão suas "condições de limite", juntamente com os casos de uso de alto nível.

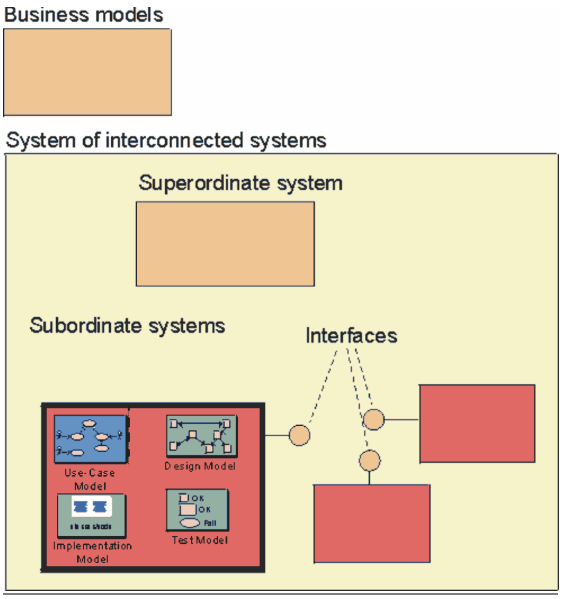

Figura 8. Os sistemas subordinados são descritos por seus próprios conjuntos de modelos.

Para mostrar como você trabalharia com o sistema subordinado, aqui estão dois planos de iteração de amostra a partir do seu ciclo de vida.

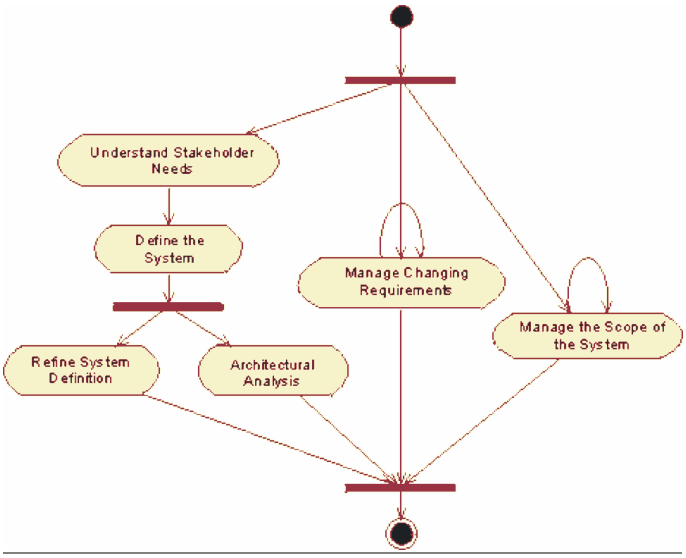

Figura 9. Um plano de iteração de iniciação de amostra para o sistema subordinado. Esta é uma iteração incompleta, já que nenhum executável é produzido.

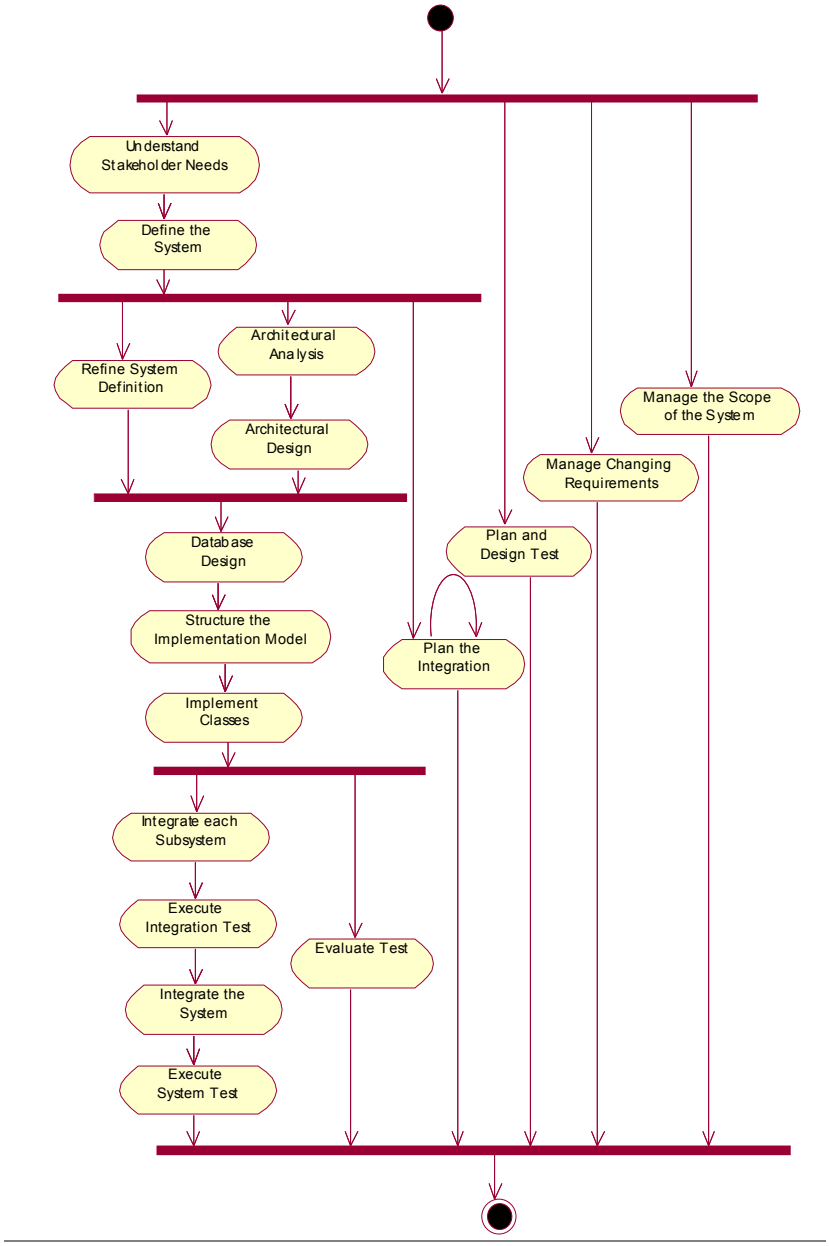

Figura 10. Um plano de iteração de elaboração de amostra para o sistema subordinado. A ênfase na elaboração está na conclusão da definição do sistema refinado e na arquitetura.

Casos de Uso em Sistemas de Sistemas Interconectados

Você deve construir um modelo de caso de uso para cada um dos sistemas, subordinador e subordinado, nos seus sistemas de sistemas interconectados. Eles são dependentes da seguinte maneira (consulte também a figura 11):

- Um caso de uso de alto nível no sistema subordinador é dividido (nem sempre, mas freqüentemente) em subsistemas. Cada 'divisão' se torna um caso de uso no modelo para o seu sistema subordinado, consulte a figura 11.
- A partir da perspectiva de um sistema subordinado, os outros sistemas subordinados são atores em seu modelo de caso de uso, consulte a figura 12.

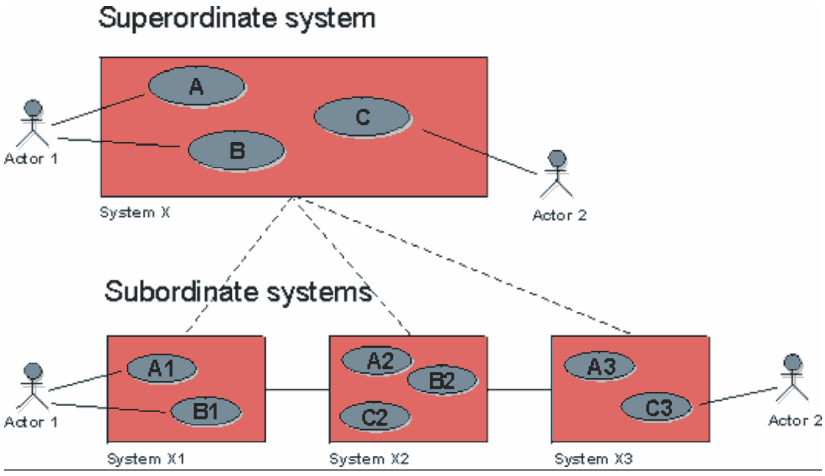

Figura 11. O relacionamento entre o caso de uso de alto nível no sistema subordinador e os casos de uso detalhados nos sistemas subordinados.

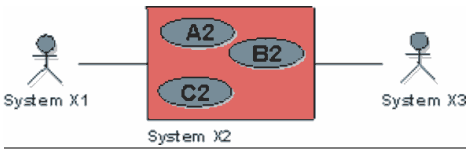

Figura 12. No modelo de caso de uso do sistema subordinado X2, os outros sistemas subordinados X1 e X3 são visualizados como atores.

Há algumas considerações especiais para os casos de uso, descrevendo o sistema subordinador. Já que você irá redescrever todos os requisitos para cada sistema subordinado, não há motivo para aprofundar-se muito em detalhes desses casos de uso. No caso normal, normalmente é suficiente somente escrever o contorno passo a passo para o fluxo de eventos dos casos de uso de alto nível e não detalhá-lo para o texto narrativo.

Neste modelo de caso de uso, você não deve utilizar qualquer um dos relacionamentos de caso de uso (generalização, estender, incluir). Em geral, não inclui valor pelos seguintes motivos:

- Você não descreverá os casos de uso de alto nível detalhadamente, então você não precisa se preocupar com o texto que aparece em vários lugares.

- De qualquer forma, você irá estruturar as informações ao "dividir" os casos de uso de alto nível em sistemas subordinados. Misturar isso com outros mecanismos de estruturação pode ser confuso.

Há uma exceção importante nisso se você pretender localizar componentes reutilizáveis no seu sistema de sistemas interconectados. Estruturar o modelo de caso de uso subordinador para localizar casos de uso genéricos é um método poderoso para localizar componentes reutilizáveis. Consulte [6] para obter detalhes adicionais sobre este tópico.

Modelos de Design nos Sistemas de Sistemas Interconectados

Cada sistema no seu sistema de sistemas interconectados, subordinador e subordinado, deve ter seu próprio modelo de design. Os modelos de design são relacionados da seguinte maneira:

- Os subsistemas no modelo de design para o sistema subordinador definem os limites dos sistemas subordinados.
- As operações que você define nos subsistemas do sistema subordinador são entradas para definir interfaces para os sistemas subordinados.

O modelo de design do sistema subordinador é descrito menos detalhadamente do que os modelos de design subordinados. Você produziria o seguinte:

- Subsistemas, descritos brevemente.
- AS realizações de caso de uso, em termos de como os subsistemas colaboram. A maneira comum de documentar essas realizações de caso de uso de alto nível é desenhar diagramas de seqüência. É produzindo esses diagramas que você define a "divisão" de um caso de uso de alto nível em sistemas subordinados, consulte a figura 13.
- Operações dos subsistemas.
- Definições de interface para os subsistemas.

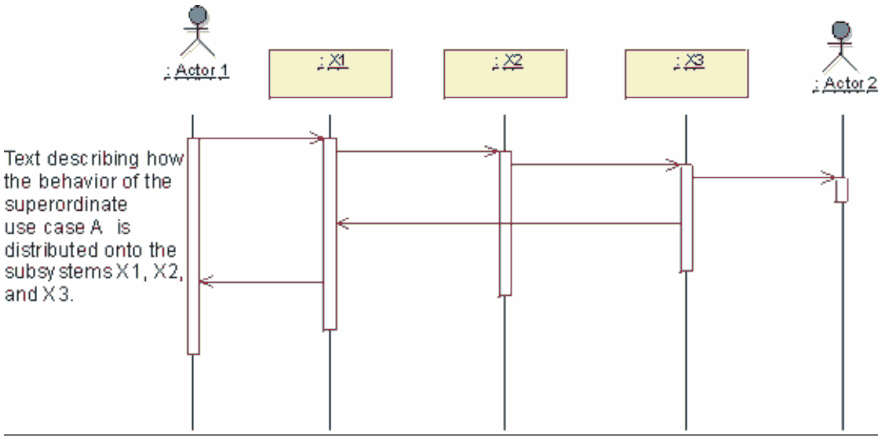

Figura 13. Um diagrama de seqüência para a realização do caso de uso subordinador A.

Conjuntos de Informações em Sistemas de Sistemas Interconectados

Uma área em que a maioria das organizações gasta muito esforço é no entendimento de como gerenciar seus artefatos e entender corretamente suas dependências. Discutimos especificamente na seção anterior a dependência entre os modelos de caso de uso e modelos de design subordinadores e subordinados. Há também problemas de dependência geral que precisam ser considerados.

Quando um sistema passar por uma transmissão do ciclo de vida, você produzirá artefatos que podem ser organizados em conjuntos de informações [8], consulte a figura 14. Esses conjuntos são organizados com base no que os artefatos desenvolvem "juntos".

- Você pode personalizar o conteúdo exato de cada conjunto, dependendo de que tipo de aplicativo você está construindo, mas os conjuntos permanecem os mesmos.
- Você precisa entender as dependências entre os conjuntos para que você possa manter a rastreabilidade entre os artefatos de um modo efetivo.

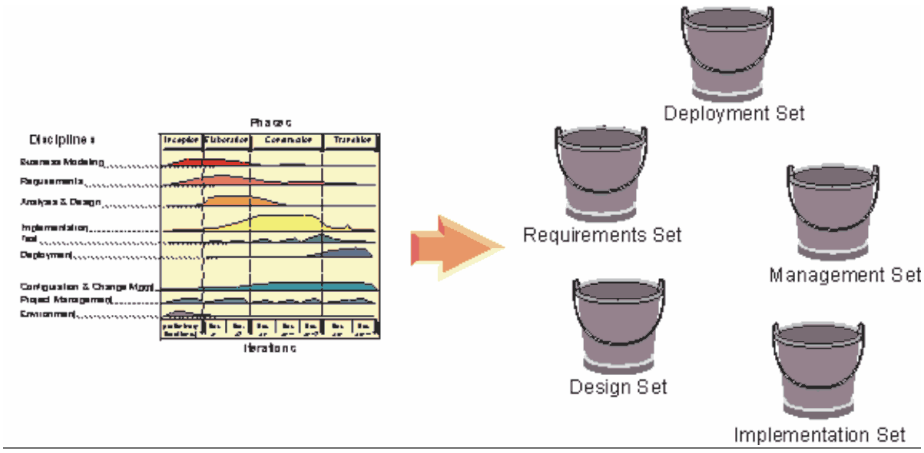

Figura 14. O ciclo de vida de um sistema produzirá conjuntos de informações.

Em um sistema do sistema interconectado, o subordinador e cada sistema subordinado produzirá produzirá seu próprio conjunto de conjuntos de informações, consulte a figura 15.

- Um conjunto de informações subordinadas tem uma dependência do seu conjunto de informações subordinadoras.
- O tipo de conteúdo pode ser diferenciado nos conjuntos de informações correspondentes entre os sistemas subordinados, já que os tipos de aplicativos podem ser diferentes.
- O conjunto de informações subordinadas correspondente deve ser independente, exceto se preencherem as mesmas interfaces do subsistema definidas no sistema subordinador.

O esforço colocado em manter a rastreabilidade entre os artefatos no sistema subordinador e os sistemas subordinados deve ser mantido no mínimo. Manter a rastreabilidade dentro do sistema deve ser prioridade.

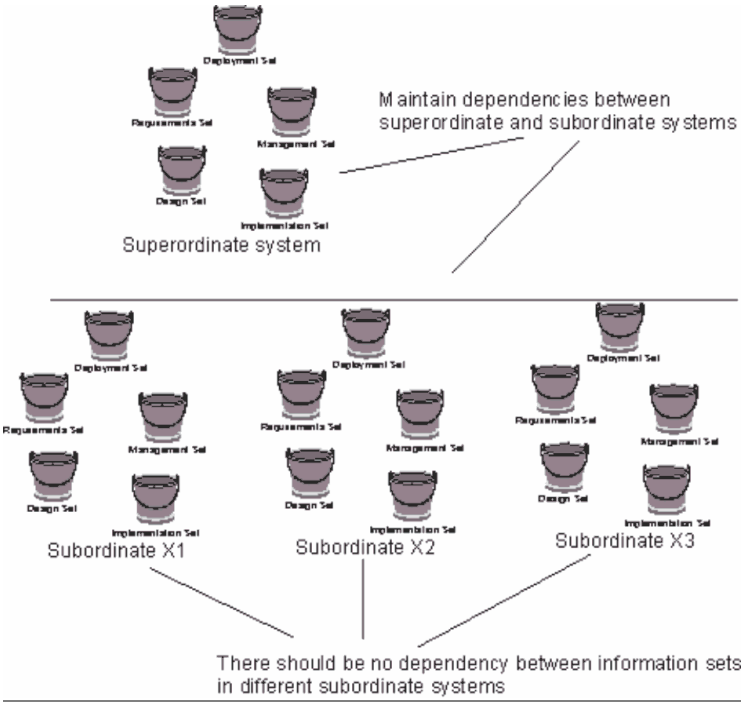

Figura 15. Cada sistema em um sistema de sistemas interconectados produzirá seu próprio conjunto de conjuntos de informações.

Arquitetura em Sistemas de Sistemas Interconectados

Cada sistema nos seus sistemas de sistemas interconectados, subordinadores e subordinados, deve ter sua arquitetura definida. Para o sistema subordinador, um documento de arquitetura deve discutir:

- Os casos de uso principais ou os cenários para o sistema subordinador.
- Camada do sistema de sistema interconectado.
- Como manipular a reutilização entre os sistemas subordinados e o que reutilizar.
- Os mecanismos principais e suas implementações, genéricos o suficiente para serem utilizados por todos os sistemas subordinados. Por exemplo, todos os sistemas subordinados devem utilizar mecanismos comuns para a comunicação, relatório de erros e gerenciamento de falhas ou o sistema subordinador não se comportará como um sistema homogêneo.

Para os sistemas subordinados, um documento de arquitetura deve esclarecer:

- A função do sistema subordinado dentro do sistema de sistemas interconectados.
- Os casos de uso principais ou os cenários para o sistema subordinado.
- Como o sistema subordinado utilizará a estrutura em camadas definida para o sistema do sistema interconectado. Outra maneira de dizer isso é que você precisa definir como o sistema subordinado preencherá a função que foi definida para ele na arquitetura em camada do sistema subordinador.

- Quais mecanismos de chave genérica serão utilizados e como, e quais mecanismos de chave específicos para aplicativos serão incluídos.
- Como a reutilização será aplicada. Especificamente, quais subsistemas serão comuns em dois ou mais sistemas subordinados e quais mecanismos serão construídos para permitir que os sistemas subordinados se comuniquem.

Relações entre Sistemas

Você tem visto que as atividades de desenvolvimento do sistema também podem ser aplicadas nos sistemas implementados pelos sistemas de sistemas interconectados. Isso é vantajoso porque significa que você não precisa manipular tais sistemas de uma maneira significantemente diferentes do que foi utilizado com outros sistemas. Você também obtém uma boa separação do sistema subordinador a partir da sua implementação na forma de outros sistemas subordinados. Cada sistema em um sistema do sistema interconectado possui seu próprio ciclo de vida. Já que cada sistema pode ter características diferentes, você pode utilizar variações do processo de desenvolvimento para produzi-las. Nos termos do Rational Unified Process [2], você teria um caso de desenvolvimento diferente para cada sistema.

Uma nota final na independência entre os sistemas envolvidos em um sistema dos sistemas interconectados:

Primeiramente, verifique os sistemas subordinados. Cada sistema implementa um subsistema no modelo de design do sistema subordinador. Os subsistemas dependem das interfaces uns dos outros e não explicitamente uns dos outros, consulte a figura 12. Assim, você pode trocar um subsistema por uma nova versão dele sem afetar outros subsistemas, desde que o novo subsistema esteja de acordo com a mesma interface. Você obtém exatamente a mesma relação entre os sistemas subordinados. Cada sistema subordinado visualiza suas proximidades como um conjunto de interfaces. Isso significa que você pode trocar um sistema por outro, desde que o novo sistema reproduza as mesmas funções em relação a outros sistemas, por exemplo, desde que possa ser representado com o mesmo conjunto de interfaces. Os sistemas referem-se às interfaces uns dos outros conforme especificado pelas relações correspondentes entre subsistemas e interfaces no modelo subordinador.

No modelo de caso de uso de um sistema subordinado, as interfaces dos outros sistemas subordinados com as quais ele interage são representadas por atores. É possível dizer que um sistema subordinado verifica as interfaces de outro sistema conforme oferecido pelos atores correspondentes e, portanto, nunca tem que se referir diretamente ao outro sistema, consulte a figura 16. Observe que a interface B ocorre em vários locais na figura 12, indicando que é realmente a mesma interface utilizada como referência pelos subsistemas no sistema subordinador e pelos sistemas subordinados correspondentes.

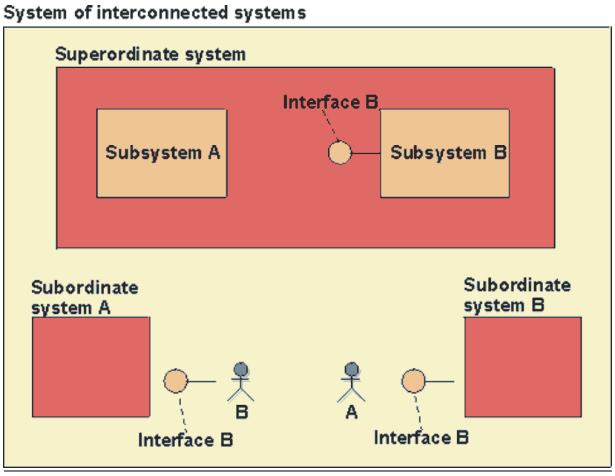

Figura 16. Os subsistemas do sistema subordinador dependem uns dos outros somente por meio de suas interfaces. Os sistemas subordinados de implementação obtêm o mesmo tipo de independência. No modelo do sistema subordinador, o subsistema B fornece a interface B para outros subsistemas. Portanto, o sistema B subordinado correspondente precisa fornecer a mesma interface B para outros sistemas subordinados.

Quanto ao sistema subordinador, qual é a sua relação com os sistemas subordinados? É independente dos seus sistemas de implementação no seguinte sentido: Cada sistema é somente uma implementação do que temos especificado nos modelos do sistema subordinador, não faz parte da sua especificação. Por motivos práticos, você deve definir os links de rastreabilidade entre os sistemas em níveis diferentes para rastrear requisitos e a maneira mais "organizada" de fazer isso é definir tais links somente entre as interfaces, consulte a figura 11. Na verdade, alguém pode até dizer que os sistemas subordinados são nada mais que implementações fornecendo as interfaces definidas nos modelos subordinadores.

Mas isso é para os sistemas que são nada mais que simples exemplos insuficientes. Uma interface não especifica nada além do que ocorre em um ponto de interação específico. Um sistema subordinado pode ter centenas de interfaces e cada interface pode ter dezenas de operações. Relacionar uma entrada em uma interface para uma ou muitas saídas em outra interface é impraticável em uma descrição da interface. Isso ocorre porque você precisa que os casos de uso expliquem a semântica do sistema subordinado.

É possível concluir que cada sistema envolvido quando um sistema é implementado por um sistema de sistemas interconectados é independente de outros sistemas, mas depende muito da interface um do outro. Isso fornece uma plataforma muito boa para o desenvolvimento paralelo dos sistemas subordinados.

Áreas de Aplicação

As técnicas de arquitetura e de modelagem para sistemas de sistemas interconectados podem ser utilizadas para tipos diferentes de sistemas, como:

- sistemas distribuídos
- sistemas muito grandes ou complexos
- sistemas combinando várias áreas de negócios
- sistemas reutilizando outros sistemas
- desenvolvimento distribuído de um sistema

A situação também pode ser inversa: a partir de um conjunto de sistemas já existentes, definimos um sistema de sistemas interconectados montando os sistemas. Na verdade, em alguns casos isso é como o sistema grande é desenvolvido nas fases iniciais de sua evolução. Você percebe que possui sistemas que poderiam ser interconectados e fazendo isso, cria um "sistema grande" que inclui mais valor do que dois sistemas separados.

Na verdade, para qualquer sistema em que é possível visualizar partes diferentes do sistema como sistemas próprios, é recomendável defini-lo como um sistema dos sistemas interconectados. Mesmo que seja um sistema único hoje, posteriormente pode ser necessário dividir o sistema em vários produtos separados por causa do desenvolvimento distribuído, os motivos de reutilização ou as necessidades dos clientes para comprar somente partes dele, esses são alguns exemplos.

Finalmente, verificaremos melhor alguns casos em que a arquitetura para os sistemas de sistemas interconectados pode ser utilizada. Mostraremos, para cada um dos exemplos, que o sistema em questão tem que ser considerado, *ambos* como um sistema único *e* como um conjunto de sistemas separados, indicando que deve ser tratado como um sistema subordinador implementado por um sistema de sistemas interconectados.

Sistemas em Larga Escala

A rede telefônica é provavelmente o maior sistema de sistemas interconectados do mundo. Este é um exemplo excelente em que mais de dois níveis de sistema são necessários para gerenciar a complexidade. Também é um exemplo de um caso em que o sistema subordinador de nível superior é de propriedade de um corpo de padronização e empresas diferentes em competição desenvolvem um ou vários sistemas subordinados que devem estar de acordo com esse padrão. Aqui discutiremos a rede de telefonia móvel GSM (Global System of Mobile Telephony) para mostrar as vantagens de implementar um sistema de larga escala como um sistema dos sistemas interconectados.

A funcionalidade de um sistema muito grande normalmente combina várias áreas de negócios. Por exemplo, o padrão GSM cobre todo o sistema a partir do assinante de chamada para o assinante chamado. Em outras palavras, inclui ambos os

comportamentos dos telefones móveis e os nós de rede. Diferentes partes do sistema são produtos próprios que são comprados separadamente, mesmo por tipos diferentes de clientes, por isso elas devem ser tratadas como sistemas próprios. Por exemplo, uma empresa que desenvolve sistemas GSM completos venderá os telefones móveis para assinantes e nós de rede para operadores de telefone. Esse é um dos motivos para tratar partes diferentes de um sistema GSM como sistemas subordinados diferentes. Outro motivo é que demoraria muito para desenvolver um sistema grande e complexo como GSM como um único sistema; as partes diferentes devem ser desenvolvidas em paralelo por várias equipes de desenvolvimento.

Por outro lado, devido ao fato de que o padrão GSM cobre todo o sistema, há motivo para também considerar o sistema como um todo, ou seja, o sistema subordinador. Isso ajudará os desenvolvedores a entenderem o domínio do problema e como partes diferentes estão relacionadas umas com as outras.

Sistemas distribuídos

Para os sistemas distribuídos em vários sistemas do computador, a arquitetura para sistemas de sistemas interconectados é muito adequada. Por definição, um sistema distribuído sempre consiste em pelo menos duas partes. Esses sistemas são muito bem agrupados também para serem *desenvolvidos* de um modo distribuído, ou seja, por várias equipes de desenvolvimento autônomo trabalhando em paralelo. Os sistemas subordinados de um sistema distribuído podem até mesmo serem vendidos como produtos próprios. Assim, é natural considerar um sistema distribuído como um conjunto de sistemas separados.

Os requisitos de um sistema distribuído normalmente cobrem a funcionalidade de todo o sistema e, às vezes, as interfaces entre as partes diferentes não são predefinidas. Além disso, se o domínio do problema for novo para os desenvolvedores, eles têm que considerar, primeiramente, a funcionalidade de todo o sistema, sem levar em consideração como será distribuído. Esses são dois motivos muito importantes para visualizá-lo como um sistema único.

Reutilização de Sistemas Legados

Freqüentemente os sistemas grandes reutilizam os sistemas legados. O legado pode ser descrito como um sistema subordinado. Uma "reengenharia" seria utilizada em um modelo de caso de uso e talvez um modelo de análise para o sistema legado para entender como pode funcionar no contexto maior do sistema subordinador. Esses modelos em reengenharia não necessariamente precisam estar completos, no mínimo eles precisam cobrir a funcionalidade do legado que possui um impacto direto na funcionalidade do restante do sistema de sistemas interconectados ou que pode requerer modificação.

Utilização de Pacotes Pré-fabricados

Um sistema pode ser uma integração e personalização de dois ou mais pacotes pré-fabricados. Um bom exemplo são os sistemas ERP (Enterprise Resource Planning). Muitos sistemas ERP são uma composição de sistemas subordinados como o MRP (Material Resource Planning), o Gerenciamento de Inventário, o Gerenciamento da Cadeia de Suprimentos, etc. as composições semelhantes estão disponíveis em outras áreas como recursos humanos ou aplicativos de folha de pagamento. São como sistemas pré-fabricados que você tem que especializar e interconectar com outros pacotes padrão para obter um sistema completo. Para entender o que o conjunto de pacotes faz, você precisa do sistema subordinador. Esse caso é o que muitos clientes na comunidade financeira enfrentam hoje.

Sumário

Esse artigo apresenta um padrão de arquitetura para sistemas de sistemas interconectados. Essa construção permite a recursão não apenas dentro de um modelo; ela considera cada subsistema um sistema isolado e a recursão está entre todos os conjuntos de artefatos de cada sistema. A arquitetura apresentada é usada para sistemas que são implementados por vários sistemas de comunicação. Cada sistema envolvido é descrito pelo seu próprio conjunto de modelos, separado dos modelos de outros sistemas.

As vantagens em utilizar essa técnica são óbvias: você pode abordar problemas mais complexos e entendê-los utilizando uma técnica de "dividir e conquistar". No entanto, a desvantagem é que você arrisca mais planejamentos elevados, não sincronizados. Temos visto exemplos de que as organizações acham muito difícil empregar um ciclo de vida iterativo no sistema subordinador por meio da execução dos riscos que estão sendo enviados para o fim do ciclo de vida do sistema subordinador. Você também precisa observar que uma estratégia de reutilização razoável e efetiva é seguida para evitar o desenvolvimento de um conjunto de sistemas "tubo de aquecimento".

Os exemplos fornecidos ilustram que a arquitetura para os sistemas de modelagem dos sistemas interconectados é útil em muitas áreas de aplicação diferentes. Na verdade, você pode utilizar a arquitetura sugerida para qualquer sistema em que seja possível visualizar as partes diferentes como sistemas próprios.

Referências

[1] Jacobson, I.; Palmkvist, K.; e Dyrhage, S., *Systems of Interconnected Systems*, ROAD, 2(1), 1995.

[2] *Rational Unified Process* versão 5.1.

[3] Rumbaugh, J.; Booch, G.; Jacobson,I., *UML Reference Manual*, Addison Wesley Longman, 1999.

[4] Herbert A. Simon, *The Sciences of the Artificial*, MIT Press, 1981.

[5] Jacobson, I.; Bylund, S.; Jonsson, P., *Using Contracts and Use Cases to Build Plugable Architectures*, Journal of Object-Oriented Programming, Maio/Junho, 1995.

[6] Jacobson, J.; Griss, M.; Jonsson, P., *Software Reuse – Architecture, Process and Organization for Business Success*, Addison Wesley Longman, 1997.

[7] Jacobson, I., *Use Cases in Large-Scale Systems*, ROAD, 1(6), 1995.

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