VM-Flow - Centro de Negócios Virtual baseado em Políticas de Interação e em Orquestração e Coreografia de Serviços

Ivo José Garcia dos Santos

Dissertação de Mestrado

Instituto de Computação
Universidade Estadual de Campinas

VM-Flow - Centro de Negócios Virtual baseado em Políticas de Interação e em Orquestração e Coreografia de Serviços

Ivo José Garcia dos Santos

Fevereiro de 2004

Banca Examinadora:

- Professor Dr. Edmundo Roberto M. Madeira (Orientador)
- Professor Dr. Manuel de Jesus Mendes - FEEC - Unicamp, UNISANTOS
- Professor Dr. Rogério Drummond - IC - Unicamp
- Professora Dra. Maria Beatriz Felgar de Toledo - IC - Unicamp (Suplente)

VM-Flow - Centro de Negócios Virtual baseado em Políticas de Interação e em Orquestração e Coreografia de Serviços

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Ivo José Garcia dos Santos e aprovada pela Banca Examinadora.

Campinas, 09 de Março de 2004.

Professor Dr. Edmundo Roberto M. Madeira
(Orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

"Porque o SENHOR dá a sabedoria,
e da sua boca vem a inteligência e o entendimento."
Provérbios 2:6

"Feliz o homem que acha sabedoria, e o homem que adquire conhecimento porque melhor é o lucro que ela dá do que o da prata, e melhor a sua renda do que o ouro mais fino. Mais preciosa é do que pérolas, e tudo o que podes desejar não é comparável a ela."
Provérbios 3:13-15

"A sabedoria é um adorno na prosperidade e um refúgio na adversidade."
Aristóteles

"Die Naturwissenschaft ohne Religion ist lahm, die Religion ohne Naturwissenschaft ist blind."
Albert Einstein

Resumo

O crescimento da Internet e da *World Wide Web*, juntamente com novos avanços tecnológicos, tem forçado as companhias a considerar com mais seriedade os recursos oferecidos pelos mecanismos de *e-Commerce* e *e-Business*. As oportunidades para cooperação entre companhias através de Empresas e Organizações Virtuais são baseadas em redes abertas e em Tecnologias de Informação e Comunicação (TICs) inovadoras. As Empresas Virtuais oferecem, de um lado, a possibilidade de compartilhar processos de negócio importantes de forma lucrativa e, de outro lado, o acesso às capacidades e recursos oferecidos pelos outros parceiros. Isto pode levar a uma redução dos ciclos de desenvolvimento e manufatura, dos custos de operação, possibilitar uma operação em alcance global e ainda permitir uma rápida adaptação às necessidades de mercado [Ouz01].

Esta dissertação apresenta uma infra-estrutura para um Centro de Negócios Virtual (*Virtual Marketplace*) que suporta *Empresas Virtuais Dinâmicas* e é baseado em *workflow*. Uma série de políticas de interação voltadas para garantir diferentes níveis de autonomia e privacidade às empresas participantes é apresentada e discutida. Além disso, Orquestração e Coreografia de Serviços são mostradas como uma solução adequada na implementação destas interações interorganizacionais baseadas em políticas. Com o objetivo de validar o modelo, um protótipo da plataforma é descrito e analisado, juntamente com uma aplicação-exemplo construída sobre ela.

Abstract

The advances of the Internet and the World Wide Web (Web) in accordance to new technological advances urged companies to seize the opportunities offered by e-Commerce and e-Business. The opportunities for co-operation among companies in terms of Virtual Communities and Virtual Enterprises are based on open networks and innovative Information and Communication Technologies (ICTs). Virtual Enterprises provide, from one side, the possibility to share key business processes in a profitable way and, from the other side, access to capabilities and resources offered by other partners. This can lead to shorten development and manufacturing cycles, reduced time to market and operational costs, global operation and reach, and rapid adaptation to new market needs [Ouz01].

This dissertation presents a Virtual Marketplace infrastructure, called VM-Flow, which supports Dynamic Virtual Enterprises and is workflow-based. A set of interaction policies that implement different autonomy and privacy levels for the marketplace partners is introduced and discussed. Also, Orchestration and Choreography of Services are shown as a suitable solution to implement these policy-based inter-organizational interactions. In order to validate the proposed model, a developed prototype is presented, together with an application built over it.

Agradecimentos

Primeiramente a **Deus**, Criador do Universo e Pai de toda a Sabedoria e Conhecimento. Obrigado Senhor pela vida e por todas as bênçãos até hoje recebidas.

Ao meu orientador **Professor Doutor Edmundo R. M. Madeira**, pelo seu apoio, incentivo, equilíbrio, ajuda e compreensão. Este trabalho seria impossível sem a sua participação.

Aos meus **Pais** por todo amor e suporte nestes meus 24 anos de vida e 20 anos de estudos.

Aos meus **Tios** pelo carinho e pelo apoio nos momentos mais difíceis.

À **Tati** pela compreensão, carinho e companheirismo nestes dois anos de Mestrado.

A todos os meus **Familiares**. Aos meus queridos **Amigos** e **Professores** de Santos e de Campinas. Todos vocês têm parte nesta conquista.

Ao **Instituto de Computação**, seu corpo docente, funcionários e colegas por todo o apoio concedido.

À **CAPES** que tornou este trabalho possível através de Bolsa de Mestrado.

Finalmente, à minha querida **UNICAMP**, que em 1997 me recebeu como aluno de Graduação em Engenharia de Computação, e que nestes 7 anos de estudo tem me proporcionado muitas conquistas, desafios e momentos de alegria.

Lista de Acrônimos

B2C Business-to-Consumer

BPEL4WS Business Process Execution Language for Web Services (ou BPEL)

BPWS4J Business Process Execution Language for Web Services Java Run Time

CORBA Common Object Request Broker Architecture

DNS Domain Name Service

DTD Document Type Definition

DVE Dynamic Virtual Enterprise

DVEC Dynamic Virtual Enterprise Coordinator

EJB Enterprise Java Beans

EV Empresa Virtual

EVE Empresa Virtual Estática

EVD Empresa Virtual Dinâmica

HTML Hyper Text Markup Language

HTTP Hyper Text Transfer Protocol

ICT Information and Communication Technologies

IETF Internet Engineering Task Force

IIOP Internet Inter-ORB Protocol

JMS Java Message Service

MPCI MarketPlace Customer Interface

MPRS MarketPlace Repository Set

RMI Remote Method Invocation

SGML Standard Generalized Markup Language

SGWF Sistema de Gerenciamento de WorkFlow

SOAP Simple Object Access Protocol

SOC Service Oriented Computing

TI Tecnologia de Informação

TIC Tecnologia de Informação e Comunicação

UDDI Universal Description, Discovery and Integration

URI Uniform Resource Identifier

VBM Virtual Business Manager

W3C World Wide Web Consortium

WfMC Workflow Management Coalition

WSCI Web Services Choreography Interface

WSDL Web Services Description Language

XML eXtensible Markup Language

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

1.1 Motivação

No mercado globalizado do mundo atual, cada vez mais as empresas têm a necessidade de encontrar novas formas de se tornarem mais competitivas. Estas têm reconhecido as necessidades de redução dos ciclos de desenvolvimento e manufatura e também dos custos operacionais, da melhora na satisfação dos clientes, de operar em escala global e também de se adaptar rapidamente às mudanças do mercado. A automatização, a colaboração e a distribuição são respostas das empresas a tais necessidades [AHK+96, Ouz01]. Como resultado, os sistemas de informação em muitas das empresas de médio a grande porte atuais apresentam uma enorme diversidade. Os avanços nas telecomunicações, redes abertas como a Internet, plataformas e tecnologias de objetos e componentes distribuídos e interoperáveis como *CORBA* [OMG02] e *Enterprise Java Beans* (EJB), além de metalinguagens como XML (Seção 3.2.2), têm permitido novas oportunidades na condução de negócios eletrônicos [Ouz01].

Empresas Virtuais

Um novo modelo de negócios, no qual as companhias são levadas a cooperar em diferentes fases do desenvolvimento de um produto e a compartilhar processos de negócio críticos, recursos, competências, habilidades e conhecimento uma com a outra, promoveu o aparecimento das *Empresas Virtuais* [OT99, CMA99]. O objetivo inicial dos sistemas de negócio baseados em empresas virtuais era permitir a distribuição de processos de negócio entre diferentes parceiros, como forma de aumentar a eficiência dos serviços já existentes, diminuindo seu custo e também permitindo a adaptação a novas mudanças no mercado [SRKT00, Ouz01].

A medida que as companhias introduziram novos sistemas de *e-Business*, as novas pos-

sibilidades destes sistemas começaram a ser vislumbradas. A partir de uma coordenação mais próxima do trabalho entre fornecedores e manufaturas, aumentos significativos na eficiência dos processos e também na produtividade foram observados. Conforme as barreiras na comunicação diminuem, os negócios se tornam capazes de ingressar em muitos tipos novos de relacionamentos, que criam possibilidades adicionais para a participação em comunidades de comércio e negócios virtuais, transformando as cadeias de valor (*value chains*) tradicionais em redes de valor (*value networks*) [DH98].

Diferentes definições são dadas para Empresas e Organizações Virtuais. No contexto deste trabalho, a seguinte definição será adotada [Ouz98]: "*uma Empresa Virtual é uma rede de diferentes domínios de negócio administrativos que cooperam compartilhando processos de negócio e recursos para prover um serviço com valor agregado ao cliente*". O Capítulo 2 desta dissertação apresenta com mais detalhes as Empresas Virtuais, suas diferentes categorias e suas características principais.

As empresas virtuais não representam um conceito novo na área de administração de empresas. Algumas grandes indústrias, especialmente na área automobilística, possuem diversos tipos de relacionamento com seus fornecedores e clientes. Estes relacionamentos "virtuais" permitem o compartilhamento de recursos e de processos de negócio. Entretanto, o nível de integração e as TICs usadas nestes relacionamentos varia muito, sendo que a maior parte das atividades ainda é executada de forma manual, *ad-hoc* [Ouz01, F96]. Neste cenário, as empresas virtuais representam uma proeminente solução para realizar estes processos de integração de forma mais dinâmica e objetiva.

Interações Interorganizacionais

No contexto atual de interconectividade eletrônica, os conceitos de automação de processos em uma organização isolada precisam ser estendidos para suportar a cooperação entre clientes e parceiros através das barreiras da organização. Os padrões de *workflow* atuais oferecem suporte limitado para esta interconectividade. Estes padrões pecam na ausência do tratamento de aspectos como coordenação e composição dinâmica de processos [SO01].

A abordagem proposta nesta dissertação, baseada em Composição Dinâmica de Serviços Web e em uma série de políticas de Interação com diferentes níveis de autonomia e privacidade representa, dentro dos limites deste trabalho, uma solução para estas questões.

O Paradigma de Computação Orientada a Serviços

O *framework* dos Serviços Web tem o objetivo de tornar-se uma implementação baseada em padrões do paradigma de Computação Orientada a Serviços (SOC - *Service Oriented Computing* - Seção 3.2). Este paradigma surgiu em resposta às mudanças ocorridas na forma como as empresas passaram a conduzir seus negócios [CKM+03].

Empresas totalmente integradas estão sendo substituídas por redes de negócio nas quais cada participante oferece aos demais serviços especializados. Infra-estruturas de TI (Tecnologia de Informação) tradicionais que eram de posse e controle de uma única organização estão dando lugar para redes de aplicações e serviços gerenciadas por diversos parceiros de negócio. Este novo ambiente computacional define um conjunto de requisitos que distingue SOC de outros paradigmas computacionais. Para operar em um ambiente orientado a serviços, as aplicações (serviços) precisam declarar suas funcionalidades e capacidades em um formato computacional pré-definido. Com base nestas declarações, a descoberta automática de serviços, sua seleção e uso se tornam uma capacidade nativa de plataformas e aplicações SOC. Uma consequência desta descoberta dinâmica é um acoplamento bem menor entre as aplicações [CKM+03].

Os serviços se tornam então blocos básicos de construção a partir dos quais novas aplicações são criadas, sendo então a composição de serviços a principal preocupação do processo de desenvolvimento de aplicações. Uma composição combina diversos serviços seguindo um determinado padrão para alcançar um objetivo de negócio, resolver um problema científico ou ainda prover novas funcionalidades genéricas. As composições de serviços podem, elas próprias, se tornarem serviços, de forma recursiva. Naturalmente, a composição de serviços provê mecanismos para integração tanto de aplicações interorganizacionais (*business-to-business*) quanto intraorganizacionais.

O Capítulo 3 desta dissertação apresenta com mais detalhes o paradigma de Computação Orientada à Serviços e também os principais mecanismos para Composição de Serviços.

1.2 Objetivos

Este trabalho de Mestrado tem como seu principal objetivo a proposição e análise de uma infra-estrutura de *Marketplace Virtual*, chamada *VM-Flow*, que suporta Empresas Virtuais Dinâmicas (Capítulo 2).

A plataforma *VM-Flow* é baseada em conceitos de Sistemas de *Workflow*, possui controle descentralizado baseado em casos móveis e oferece diferentes níveis de privacidade e autonomia aos seus parceiros através de diversas *Políticas de Interação*. Esta políticas são implementadas aplicando-se recursos de Orquestração e Coreografia de Serviços Web.

As principais contribuições deste trabalho são:

- A proposição das Políticas de Interação, com diferentes níveis de autonomia e privacidade nas relações interorganizacionais.
- A implantação das Empresas Virtuais Dinâmicas através de recursos de Composição de Serviços Web (Orquestração e Coreografia).

- A implementação de um protótipo da plataforma *VM-Flow* e também de uma aplicação-exemplo sobre ela.

1.3 Estrutura do Trabalho

Esta dissertação está organizada de acordo com a seguinte estrutura de capítulos:

- **Capítulo 2**: Apresenta as Organizações Virtuais, em especial as Empresas Virtuais e os *Marketplaces*.
- **Capítulo 3**: Apresenta os conceitos fundamentais relacionados com Composição de Serviços, especialmente Orquestração e Coreografia.
- **Capítulo 4**: Este capítulo introduz o modelo da infra-estrutura da plataforma *VM-Flow* e também como é feito o gerenciamento das Empresas Virtuais Dinâmicas. Diferentes *Políticas de Interação* suportadas pelo *VM-Flow* são apresentadas.
- **Capítulo 5**: A implementação do núcleo da plataforma, bem como a orquestração e coreografia dos serviços nas Empresas Virtuais Dinâmicas são discutidas neste capítulo.
- **Capítulo 6**: Uma série de trabalhos relacionados aos diferentes tópicos que esta dissertação contempla são analisados. Os pontos comuns e as diferenças com relação ao *VM-Flow* também são apresentados.
- **Capítulo 7**: Algumas análises comparativas são discutidas e também as considerações finais, incluindo possíveis extensões a este trabalho, são apresentadas.

Capítulo 2

Organizações Virtuais

Neste capítulo, de caráter conceitual, são apresentadas as Organizações Virtuais, mais especificamente as Empresas Virtuais e suas diferentes categorias e também os Centros de Negócio Virtual (*Marketplaces*).

2.1 Empresas Virtuais

As Empresas Virtuais (EVs) representam um conjunto de entidades geograficamente distribuídas, que podem diferir tanto funcionalmente quanto culturalmente e que estão ligadas através de tecnologias de informação e comunicação, que compartilham competências, infra-estrutura e processos de negócio, com o propósito de atender a uma necessidade de mercado específica.

Conforme já apresentado no Capítulo 1, a definição de Empresa Virtual que será adotada no contexto deste trabalho é a seguinte [Ouz98]: "*uma Empresa Virtual é uma rede de diferentes domínios de negócio administrativos que cooperam compartilhando processos de negócio e recursos para prover um serviço com valor agregado ao cliente*". Note que nesta definição, os *domínios de negócio administrativos* representam as empresas reais que irão compor uma empresa virtual (e não um setor de mercado). Além disso, a limitação no tempo de existência de uma empresa virtual está vinculada ao término da execução de seu objetivo. Ouzonis [Ouz98] afirma também que do ponto de vista de um observador externo (por exemplo um cliente), a empresa virtual aparece como uma empresa única.

Na literatura, duas categorias de EVs são identificadas [SRKT00, Ouz01]:

- **Empresas Virtuais Estáticas**: nesta categoria, os relacionamentos entre os parceiros são estáticos, pré-configurados e não podem ser alterados facilmente.
- **Empresas Virtuais Dinâmicas**: esta categoria representa uma evolução da anterior, tirando vantagem significativa das oportunidades oferecidas pela Internet e pela

economia globalizada. As EVs Dinâmicas possuem ciclos de vida bem curtos, relacionamentos de negócio dinâmicos entre os parceiros e comportamento autônomo e flexível.

Pode-se considerar uma série de características como sendo comuns à maioria das Empresas Virtuais. São elas [OT99, OT98]:

- Os processos que compartilham do negócio são distribuídos através de parceiros distintos e separados geograficamente.
- Os parceiros podem ser autônomos, ou seja, não é necessário existir cooperação ou algum tipo de dependência fora do escopo da empresa virtual.
- Os parceiros são organizados internamente de uma maneira que eles possam delegar parte de seus processos de negócio e recursos total ou parcialmente para a organização virtual.
- Os parceiros devem fazer acordos estabelecendo seus objetivos comuns e como alcançá-los, atribuindo papéis e distribuindo responsabilidades (inclusive com relação a divisão de lucros e prejuízos).
- Os parceiros devem usar infra-estruturas comuns de comunicação e informação de modo a suportar colaboração e coordenação durante o ciclo de vida da empresa virtual.

As empresas virtuais podem ser classificadas de acordo com os seguintes critérios [CMA99]:

- **Duração**: As alianças podem ser temporárias ou de mais longa duração.
- **Formação**: A formação de uma EV pode ser dinâmica, com membros entrando e saindo a todo tempo, ou ainda estática.
- **Participação**: Define se uma organização participa ou não de múltiplas alianças simultaneamente.
- **Coordenação**: Classificação de como a aliança é coordenada, por exemplo, por uma organização principal no centro ou se o comando é distribuído ao longo das organizações.
- **Escopo de Visibilidade**: Relacionado ao quão 'longe' um nó pode enxergar na rede. Na maioria dos casos um nó enxerga somente seus vizinhos (ou parceiros) diretos. Já em outros, um nó pode ter direitos de visibilidade sobre outras empresas não diretamente relacionadas.

2.2 Categorias de Empresas Virtuais

Conforme já mencionado anteriormente neste capítulo, duas categorias de empresas virtuais podem ser definidas: *Empresas Virtuais Estáticas* e *Empresas Virtuais Dinâmicas*. Nas duas subseções a seguir estas duas categorias genéricas serão analisadas com mais detalhes e alguns exemplos serão apresentados para esclarecer melhor os diferentes modelos, conceitos, vantagens e desvantagens que cada uma das abordagens possui.

2.2.1 Empresas Virtuais Estáticas

Nas Empresas Virtuais Estáticas (EVEs), um conjunto de parceiros de negócio é ligado de forma estática e fixa, isto é, os processos de negócio que são compartilhados são fortemente integrados. Os relacionamentos entre os parceiros (as interfaces entre os processos) são pré-definidos e fortemente acoplados.

Tipos de Empresas Virtuais Estáticas

Baseado no estilo de distribuição e gerenciamento da rede, dois tipos de EVEs podem ser identificados: centralizada e descentralizada. Nas EVEs Centralizadas, um domínio de negócio é dominante e coordena os relacionamentos entre os membros da rede. Os parceiros e essa organização central formam relacionamentos de longa duração. O estabelecimento da EVE é realizado manualmente, de forma personalizada, e sob controle total da organização dominante. Exemplos típicos de EVEs Centralizadas são modelos que foram aplicados na indústria de automóveis [ZP99]. Neste caso, uma grande indústria de automóveis possui uma rede de fornecedores, distribuidores e revendedores que trabalham juntos em diferentes etapas dos processos de produção, distribuição e revenda. O fabricante possui necessidades específicas, sempre tendo como objetivo aumentar o grau de automação e diminuir os custos de produção e distribuição. A rede de fornecedores, distribuidores e revendedores cooperam de forma muito próxima com o domínio de negócios central (o fabricante) e de acordo com suas determinações.

Já nas EVEs Descentralizadas, diferentes parceiros de negócio se unem de forma mais autônoma sem um controle central. Este tipo de rede é similar à anterior exceto pelo fato de que não há organização central e dominante e que cada membro da rede pode cooperar com muitos outros domínios [Ouz98]. Nenhum dos parceiros possui controle total sobre a rede e a integração entre os processos de negócio dos membros é executada de forma conjunta. As indústrias de alta tecnologia exemplificam esse modelo. Organizações como fábricas de semi-condutores e de placas focam-se em uma atividade em uma cadeia de valor completa e se associam com múltiplas organizações de forma a participar de diversas cadeias. Neste exemplo, os membros da EVE podem trabalhar na produção e montagem

de novos produtos, bem como na distribuição de produtos para diferentes revendedores.

2.2.2 Empresas Virtuais Dinâmicas

Nas Empresas Virtuais Dinâmicas (EVDs), um conjunto de parceiros de negócio é ligado dinamicamente, por demanda, e de acordo com os requisitos dos clientes, formando um *Centro de Negócios Virtual* (ou *Marketplace* - Subseção 2.3). Os domínios de negócio não possuem relações de negócio fixas e dessa maneira a EVD pode ser alterada continuamente de acordo com critérios determinados pelo mercado [Fie98].

Os domínios de negócio que desejam formar relacionamentos em uma EVD podem registrar ofertas no *marketplace* relacionadas com um determinado *template* de um processo de negócio. A qualquer momento que um domínio de negócio necessite usar um processo particular, ele procura no *marketplace* e localiza todos parceiros potenciais que podem prover o serviço. Assim que a lista dos candidatos é formada, o processo de seleção se inicia. O processo de seleção entre os domínios é normalmente realizado através de negociação (o processo de negociação pode ser tanto manual quanto automatizado). O resultado é um contrato que regula as relações de negócio entre os domínios envolvidos.

Os *marketplaces* são, em geral, organizados em torno de certos serviços ou *templates* de produtos especificados globalmente e que podem ser oferecidos por diferentes fornecedores. Apesar dos *marketplaces* e dos mecanismos de casamento entre cliente e fornecedores já terem sido usados por algum tempo no comércio eletrônico B2C (*Business-to-Consumer*), pouca experiência se tem com relação a aplicação desse mecanismo com Empresas Virtuais Dinâmicas [OT98]. A principal razão era a falta de tecnologias que permitissem uma definição fácil e flexível de *templates* de processos, mecanismos de negociação automatizada e interação autônoma entre diferentes domínios. Com o advento de tecnologias como XML (*eXtensible Markup Language*) e sua aceitação como meta-linguagem da Internet, conceitos como o de *marketplace* ganharam força [OT99].

2.3 Centros de Negócios Virtual ou *Virtual Marketplaces*

Os *Virtual Marketplaces* representam um ambiente virtual no qual compradores e vendedores podem se encontrar e se engajar em atividades de comércio eletrônico, como compra e venda de produtos e serviços [Ouz01]. Diversos tipos de *marketplaces* foram propostos e desenvolvidos até o momento, com ênfase em diferentes aspectos (comércio eletrônico do tipo *business-to-consumer*, *business-to-business*, leilões virtuais, etc). Entretanto, as características gerais permanecem as mesmas - em geral os marketplaces oferecem três tipos de serviço [Ouz01]:

- Administração de *templates* de produtos e serviços;
- Gerenciamento do registro dos fornecedores de produtos/serviços;
- Gerenciamento dos pedidos dos compradores.

No âmbito das EVs, uma abordagem mais recente para automatizar seu processo de formação é o uso de um *marketplace* virtual ou de um serviço de diretórios no qual membros potenciais registram seus recursos e processos de negócio [Pro98]. Os centros de negócios virtuais são muito importantes para garantir a competitividade das EVDs [OT01]. Estes centros fornecem infra-estrutura e diversos tipos de serviços para estas empresas, aumentando sua flexibilidade e escalabilidade através de funções de busca e mediação e também do suporte à seleção de parceiros de negócios durante as fases de estabelecimento e reconfiguração das mesmas. Esses serviços podem ser complementados por outros mais avançados, como a negociação automatizada e contratos eletrônicos.

O *marketplace* virtual provê o casamento de serviços com domínios de negócio que desejam localizar parceiros em uma EV. Esta abordagem tira vantagem dos recursos oferecidos pela Internet e melhora significativamente o processo de formação das organizações virtuais. Os *marketplaces* podem ser usados de forma mais efetiva e dinâmica não só na fase de formação das EVs, mas também durante sua fase de execução. Isso significa que os parceiros envolvidos na realização dos processos compartilhados podem se modificar continuamente e dinamicamente de acordo com os requisitos do cliente e dos processos. Neste caso, para cada execução de um processo de negócio uma nova EV é criada de forma dinâmica de acordo com as necessidades do cliente e dos parceiros individualmente.

2.4 Ciclo de Vida das Empresas Virtuais Dinâmicas

Um modelo de ciclo de vida geralmente descreve as principais fases e atividades requeridas durante a existência de uma entidade. No contexto das Empresas Virtuais, o ciclo de vida consiste de duas fases principais que devem ser seguidas para a formação e gerenciamento das EVDs. Cada uma dessas fases é formada por passos mais específicos que descrevem as suas principais operações.

2.4.1 Cenário

De forma a um melhor entendimento do ciclo de vida das EVDs, o seguinte cenário é usado como exemplo: uma empresa chamada *ABC Projetos e Engenharia* vende *on-line* projetos de casas em condomínios de alto padrão. Esse processo inclui várias etapas, dentre elas a elaboração de um projeto (vinculado a um engenheiro e a um arquiteto) a

partir das necessidades do cliente, a aprovação do projeto pelo cliente, a compra de materiais de construção de diversos fornecedores e, em alguns casos, a compra dos ítens que irão compor a decoração da casa. A *ABC Projetos e Engenharia* na verdade gerencia o projeto, terceirizando as diversas etapas que o compõem. De forma a encontrar parceiros potenciais, a *ABC* utiliza os serviços de um *marketplace* virtual especializado na área de construção civil. Esse *marketplace* virtual especifica *templates* para diversos processos - um desses *templates* está relacionado, por exemplo, com o processo de compra de materiais de construção. As empresas que vendem materiais de construção usam esse *template* padrão e registram suas ofertas de serviço no *marketplace*. Quando um cliente faz o pedido de um projeto para a *ABC*, após encontrar um engenheiro e um arquiteto (possivelmente também através do *marketplace*) e da elaboração de diversos pré-projetos, a companhia procura no *marketplace* os parceiros potenciais para o fornecimento dos materiais necessários, negocia com eles os parâmetros relacionados com o pedido e então seleciona os mais adequados para cada pré-projeto. Uma vez escolhido um desses pré-projetos pelo cliente, os fornecedores previamente escolhidos são contatados para a efetivação do pedido. Interações similares com o *marketplace* ocorrem nas demais etapas do processo de venda e construção da casa. A seleção dos parceiros está fortemente associada com as características do projeto escolhido, portanto, quando um novo cliente vem e faz um novo pedido na *ABC*, a companhia usa novamente o *marketplace* e localiza, negocia e seleciona provavelmente outros parceiros que podem melhor atender os requisitos desse novo cliente.

O cenário que acaba de ser descrito é, apesar de simplificado, bem típico, revelando algumas das características das EVDs. De forma a suportar um cenário como esse, uma plataforma de gerenciamento de EVDs é necessária [CMA99, SRKT00].

2.4.2 Algumas considerações

Inicialmente, quando diferentes domínios de negócio não possuem qualquer relacionamento entre si, esses domínios são considerados candidatos a parceiros. Um candidato se torna um parceiro efetivo após um processo de negociação. Quando um acordo é alcançado, esse candidato se torna então um parceiro efetivo da EV. Esse processo de negociação é feito dinamicamente e durante o oferecimento do serviço ao cliente - o acordo pode durar por apenas uma instância do processo de negócio ou por diversas. Um domínio de negócio se torna um parceiro potencial quando ele registra os processos de negócio que pode oferecer em um *marketplace*.

2.4.3 Fases do Ciclo de Vida

De acordo com Ouzonis [OT99, OT01], o ciclo de vida das EVDs consiste das seguintes fases:

- **Fase de Especificação e Registro do Processo de Negócio**: durante essa fase os diferentes domínios de negócio devem especificar seus processos de negócio locais e remotos. A especificação dos processos de negócio é realizada através de uma linguagem de definição de processos de negócio. Em seguida, todo domínio registra para cada processo local suas ofertas correspondentes no *marketplace*, expondo as condições nas quais esses processos locais serão oferecidos a outros domínios.

- **Fase de Gerenciamento do Processo de Negócio**: durante essa fase, um domínio de negócio oferece serviços a clientes empregando o modelo dinâmico do *marketplace*. Sempre que um cliente solicita um serviço, o domínio inicia a provisão do mesmo. Se o serviço solicitado for composto por sub-processos remotos que devem ser executados por outros domínios, então o domínio de negócio aciona o *marketplace*, localizando os parceiros potenciais, negociando com eles dinamicamente e finalmente estabelecendo as parcerias para a execução do serviço. Todo esse processo é transparente ao cliente. Durante a execução do serviço, o cliente pode gerenciar o mesmo, ou seja, suspender, reiniciar ou mesmo terminar sua execução. Cada pedido de gerenciamento do cliente deve ser executado de forma autônoma, distribuída e interorganizacional.

Apresentaremos no capítulo referente ao modelo (Capítulo 4) um detalhamento maior sobre o funcionamento das EVDs no contexto deste trabalho.

Capítulo 3

Composição de Serviços

Os sistemas de *Workflow* e também o paradigma computacional de *Computação Orientada a Serviços* são discutidos neste capítulo. Neste contexto, a composição de serviços é apresentada, com destaque para os mecanismos de *Orquestração* e *Coreografia*. Especificações como SOAP, XML, WSDL, BPEL4WS e WSCI são discutidas.

3.1 Sistemas de Workflow

Workflows podem ser definidos como modelos computadorizados de processos de negócio que especificam todos os parâmetros envolvidos em sua execução [SF00]. O termo *gerenciamento de workflow* é anterior aos atuais *Sistemas de Gerenciamento de Workflow* (SGWFs). Este termo refere-se ao domínio de aplicação que envolve logística e processos empresariais. Esta área de estudo é também conhecida como logística de escritórios. O principal objetivo dos SGWFs é assegurar que atividades apropriadas sejam executadas pelas pessoas certas, no tempo correto. Apesar de ser possível realizar gerenciamento de *workflow* sem o uso de um sistema de gerenciamento de *workflow*, a maioria das pessoas associa este conceito com tais sistemas [SF00]. A *Workflow Management Coalition* (WfMC) [Hol94], entidade responsável por uma tentativa de padronização dos sistemas de *workflow*, define os *SGWFs* como sendo: "*Sistemas que definem, executam e gerenciam completamente workflows através da execução de um software cuja ordem de execução é dirigida por uma representação lógica e computadorizada de um workflow*".

Do ponto de vista empresarial, os SGWFs são usados para coordenar e seqüenciar processos de negócio (*business processes*). Tais processos são representados por um fluxo de atividades (passos) individuais (como entrada de dados do comprador, consultar banco de dados ou verificar uma assinatura), pelo estabelecimento de uma determinada ordem de condições sob as quais estes passos devem ser executados (incluindo aspectos como fluxo de dados entre as atividades), pelo desígnio de pessoas ou processos responsáveis

por cada tarefa e por aplicações que irão auxiliar no desempenho de cada atividade [SF00].

Existem diversos cenários onde SGWFs podem ser empregados: processos de desenvolvimento de software, pedidos de financiamento de casa própria, controle de manufatura, controle de empréstimos, automação de cartórios e de fóruns de justiça, acompanhamento de pacientes em hospitais, automação de processos logísticos, planejamento da produção, dentre outros [SF00].

Terminologia

Os principais termos relacionados com os SGWFs são:

- *Processo*: uma descrição de uma seqüência de tarefas ou atividades que devem ser executadas para a realização de um determinado objetivo.
- *Atividades*: representam tarefas, que são atribuídas a determinados *papéis*. Uma atividade pode ser composta por uma ou mais *(sub)tarefas*. Atividades podem ser desempenhadas de forma automática ou manual por atores ou usuários (programas ou pessoas). Uma atividade mais complexa pode ser representada por um *processo*, neste caso, chamado de *sub-processo* do processo que contém esta atividade.
- *Papel*: Função ou cargo associado a pessoas ou programas.

Algumas características dos SGWFs

De forma a coordenar e modelar as atividades de longa duração do mundo real, os SGWFs de maneira geral devem satisfazer alguns requisitos [SF00]:

- *Composição de atividades*: De forma a facilitar a reutilização e a implementação de novos processos, atividades podem ser agrupadas, compondo sub-processos que podem ser utilizados como atividades de outros processos.
- *Suporte a interoperabilidade entre diferentes SGWFs*
- *Suporte a recuperação de falhas*: O SGWF deve ser capaz de lidar com falhas de software e hardware, de forma transparente, maximizando a disponibilidade do sistema, escondendo, dentro do possível, o tratamento de erros dos usuários.
- *Monitoramento*: Ao lidarmos com processos longos e muitas vezes complexos, mecanismos que possibilitem determinar o estado corrente dos processos em execução constituem a base para ações de gerenciamento, fornecendo dados para detecção de falhas e gargalos de desempenho no sistema.

- *Auditoria*: A manutenção do histórico das execuções dos processos em SGWFs são dados importantes, servindo de base para processos de otimização, avaliação e reestruturação dos processos de Workflow de uma empresa.

- *Reconfiguração Dinâmica*: Devido à característica de longa duração de muitas atividades e à necessidade de tratamento de exceções em processos, os SGWFs devem permitir a mudança dinâmica das definições de processos em andamento. A reconfiguração dinâmica (*dynamic change*) consiste em alterar o plano de um processo enquanto suas instâncias estão em execução.

- *Tratamento de Exceções*: Exceções costumam ocorrer com certa freqüência em SGWFs. Para certas aplicações, as exceções são a regra. Por exemplo, um ator selecionado para realizar uma atividade que dura dias, pode ter que se ausentar, necessitando ser substituído durante este período.

Ao lidarmos com SGWFs distribuídos (ao contrário dos tradicionais, que possuem uma estrutura de controle centralizada), algumas características adicionais merecem destaque [SF00]:

- *Escalabilidade*: Grandes empresas como bancos e companhias aéreas envolvem milhares de usuários e centenas de milhares de processos concorrentes distribuídos em milhares de sites. Os SGWFs devem suportar o crescimento e as exigências de tais sistemas satisfazendo seus requisitos de desempenho, permitindo atender a um número cada vez maior de processos, casos e atividades simultâneas.

- *Disponibilidade*: Para que possam ser utilizados em aplicações de missão crítica, além de garantir a consistência dos dados e da aplicação, SGWFs devem prover mecanismos de tolerância a falhas (como replicação e distribuição de dados e controle) de forma a impedir que falhas no sistema paralisem os processos em andamento, suspendendo sua execução por longos períodos de tempo.

- *Segurança*: A descentralização introduz problemas de segurança. Em sistemas descentralizados, dados e controle do *workflow* são normalmente distribuídos entre máquinas (potencialmente) menos confiáveis e menos seguras que as normalmente empregadas em soluções centralizadas. Desta forma, se no caso centralizado o gerenciamento dos dados e o controle de execução eram responsabilidade de um único servidor, em *workflows* descentralizados, esta responsabilidade é compartilhada pelos nós por onde o sistema está distribuído.

3.2 Computação Orientada a Serviços

Computação Orientada a Serviços (*SOC - Service-oriented computing*) é o paradigma computacional que considera os serviços como elementos fundamentais para o desenvolvimento de aplicações. Nesse contexto, os serviços podem ser definidos como componentes abertos, auto-descritos, que suportam o desenvolvimento rápido e de baixo custo de aplicações distribuídas. Descrições são usadas para divulgar o que o serviço oferece, sua interface, seu comportamento e qualidade. Os clientes de um serviço (por exemplo, organizações que atuem como usuários-finais) e os agregadores de serviço (organizações que consolidam múltiplos serviços em um novo serviço) usam estas descrições para alcançar seus objetivos [PG03].

A aplicação de SOC na Web é manifestada através dos Serviços Web (*Web Services*). Um Serviço Web é um tipo específico de serviço que é identificado por uma URI (*Uniform Resource Identifier* [BLFM98]), cuja descrição e transporte utilizam padrões da Internet. As interações entre Serviços Web tipicamente ocorrem através de chamadas SOAP (Subseção 3.2.1) que carregam consigo conteúdo XML (Subseção 3.2.2). As descrições da interface são expressas em WSDL (*Web Services Definition Language* - Subseção 3.2.4). O padrão UDDI (*Universal Description, Discovery and Integration* - Subseção 3.2.5) define um protocolo para os serviços de diretório que contém as descrições dos Serviços Web, permitindo que clientes localizem serviços e descubram seus detalhes.

3.2.1 SOAP

O protocolo SOAP (*Simple Object Access Protocol*), recomendado pelo W3C (*World Wide Web Consortium* [W3C03b]), provê um mecanismo simples e leve para troca de informações estruturadas entre pares em um ambiente distribuído e descentralizado usando XML [BEK+00].

A grande vantagem do uso do protocolo SOAP sobre HTTP é que ele consegue atravessar redes protegidas por *Firewalls*, um problema que outros protocolos (como CORBA/IIOP ou JAVA/RMI por exemplo) têm sérias dificuldades em enfrentar.

Uma mensagem SOAP (Figura 3.1) é um documento XML comum, contendo os seguintes elementos:

- *Envelope (requerido)*: elemento que identifica o documento XML como sendo uma mensagem SOAP.
- *Cabeçalho (opcional)*.
- *Corpo (requerido)*: contém informações da chamada e de resposta.

- *Falha (opcional)*: provê informações sobre erros que ocorreram ao processar a mensagem.

A seguir, um exemplo da estrutura de uma mensagem SOAP (Figura 3.1).

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
    <soap:Header>
        ...
        ...
    </soap:Header>
    <soap:Body>
        ...
        ...
        <soap:Fault>
            ...
            ...
        </soap:Fault>
    </soap:Body>
</soap:Envelope>
```

Figura 3.1: Estrutura de uma mensagem SOAP

3.2.2 XML

A linguagem XML (*eXtensible Markup Language* [W3C00]) foi definida pelo W3C como uma linguagem de marcação que suporta a descrição de dados e cujos marcadores (*tags*) não são pré-definidos. XML descreve uma classe de objetos de dados chamados documentos XML e também descreve de certa forma o comportamento dos programas que processam esses documentos. XML (assim como HTML) é uma extensão de SGML (*Standard Generalized Markup Language*).

Algumas características importantes:

- Os documentos XML são compostos de unidades de armazenamento denominadas entidades.
- As marcações codificam uma descrição do *layout* do documento e de sua estrutura lógica.

- XML usa DTD (*Document Type Definition*) ou um XML *Schema* para descrever os dados, de forma que XML + DTD ou XML + *Schema* sejam auto-descritivos.

O exemplo na Figura 3.2 mostra um documento XML que descreve algumas características de um livro (<LIVRO>): o título (<TITULO>) e o assunto (<ASSUNTO>). Observe que este documento é bem similar a um documento HTML - a principal diferença é que os marcadores são personalizados e representam descrições dos dados marcados.

```
<?xml version="1.0"?>
<LIVRO>
    <TITULO>
     Java e XML
    </TITULO>
    <ASSUNTO>
     Processamento de documentos XML com Java
    </ASSUNTO>
</LIVRO>
```

Figura 3.2: Exemplo de documento XML - *livro.xml*

Documentos bem-formados e Documentos válidos

Um documento XML precisa ser bem-formado, ou seja, ele deve seguir as regras de sintaxe estabelecidas pela especificação XML 1.0. Algumas dessas regras:

- O documento precisa conter um ou mais elementos.
- Deve haver somente um elemento raiz, que deve conter todos os outros elementos.
- Cada elemento deve estar corretamente aninhado dentro de outro elemento, tendo o seu marcador (*tag*) de fechamento correspondente.

Um documento XML é dito válido se houver uma *Definição de Tipo de Documento* (DTD) ou um *Schema* associado a ele. Observe que um documento bem-formado nem sempre é válido, mas um documento válido é sempre bem-formado [Vel03].

3.2.3 DTD e XML Schema

Muito do sucesso crescente de XML como um formato de representação de dados se deve à possibilidade de especificar requisitos estruturais para documentos: regras que

especificam exatamente quais os tipos de conteúdo dos elementos e quais sub-elementos podem ocorrer dentro destes elementos (ordem, cardinalidade etc). A especificação XML 1.0 recomendada pelo W3C determina que essas regras, assim como em SGML, devem ser expressas através do uso de DTDs.

A Figura 3.3 mostra um exemplo de DTD associado ao documento *livro.xml* da Figura 3.2. Veja que `<!ELEMENT>` indica a definição de um novo elemento. Neste caso, `LIVRO` é composto de dois outros elementos: `TITULO` e `ASSUNTO`. Estes, por sua vez, são ambos tipo `#PCDATA` (texto).

```
<!ELEMENT LIVRO(TITULO,ASSUNTO)>
<!ELEMENT TITULO(#PCDATA)>
<!ELEMENT ASSUNTO(#PCDATA)>
```

Figura 3.3: Exemplo de trecho de DTD - *livro.dtd*

Entretanto, existem algumas características que os DTDs não conseguem tratar e que podem aparecer com freqüência. A principal limitação dos DTDs é sua simplicidade na expressão de tipos de dados: é possível especificar que um elemento deve conter PCDATA (uma *string*), mas não que ele deve conter, por exemplo, um Inteiro-não-negativo [Mer01]. Além disso, os DTDs dificultam a especificação da cardinalidade de sub-elementos: pode-se especificar de maneira compacta 'um ou mais', mas especificar 'entre 7 e 12' é, apesar de possível, excessivamente verboso.

Com o intuito de suprir essas deficiências dos DTDs é que surgiram os *XML Schemas*. A norma *XML Schema* especifica uma linguagem, baseada em XML, para a definição de esquemas de representação de dados. O propósito de um esquema XML é definir e descrever uma classe de documentos XML através da utilização dos componentes da linguagem para restringir e documentar o significado, a utilização e as relações entre as partes constituintes do documento: tipos de dados, elementos (e seu conteúdo) e atributos (e seus valores). Em resumo, um esquema XML define um vocabulário para uma classe de documentos XML.

Seu ponto mais forte é o suporte a diversos tipos de dados. Além de prover um conjunto extremamente rico de tipos simples, os *XML Schemas* permitem a derivação de novos tipos usando uma sintaxe baseada em expressões regulares. Os tipos básicos suportados são os esperados em qualquer linguagem de programação: *string*, *int*, *float*, *unsignedLong*, *byte* e assim por diante. Também inclui *timeInstant* (dia/hora), *recurringDate* (dia do ano), *uriReference*, *language* e *nonNegativeInteger*.

Veja nas Figuras 3.4, 3.5 e 3.6 um documento XML, um DTD e um *XML Schema* relacionados a um 'recado'.

```
<?xml version="1.0"?>
<recado>
    <para>Pedro</para>
    <de>Patrícia</de>
    <cabeçalho>Lembrança</cabeçalho>
    <corpo>Não me esqueça neste final de semana !</corpo>
</recado>
```

Figura 3.4: Documento XML - *recado.xml*

```
<!ELEMENT recado (para, de, cabeçalho, corpo)>
<!ELEMENT para (#PCDATA)>
<!ELEMENT de (#PCDATA)>
<!ELEMENT cabeçalho (#PCDATA)>
<!ELEMENT corpo (#PCDATA)>
```

Figura 3.5: DTD - *recado.dtd*

```
<?xml version="1.0"?>
<xs:schema>
    <xs:element name="recado">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="para" type="xs:string"/>
                <xs:element name="de" type="xs:string"/>
                <xs:element name="cabeçalho" type="xs:string"/>
                <xs:element name="corpo" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Figura 3.6: XML Schema - *recado.xsd*

Na Figura 3.5, o DTD *recado.dtd* define os elementos associados ao documento *recado.xml*: o elemento `recado` é definido como uma composição dos elementos `para`, `de`, `cabeçalho` e `corpo`. Estes por sua vez são definidos como `#PCDATA`.

Já na Figura 3.6, o *XML Schema recado.xsd* define os mesmos elementos, mas de uma forma hierárquica: `xs:element` marca a definição de um novo elemento, que no caso de `recado`, é um tipo complexo (`xs:complexType`) composto de uma seqüência (`xs:sequence`) de outros elementos (`para`, `de`, `cabeçalho` e `corpo`), todos simples e do tipo `xs:string`.

Acredita-se que os *XML Schemas* sejam os sucessores naturais dos DTDs, pois:

- São facilmente extensíveis, ao contrário dos DTDs.
- São mais poderosos, suportando diversos tipos de dados, além de permitir a criação de novos tipos compostos de maneira simples.
- São escritos em XML.
- Suportam *namespaces* (uma coleção de nomes, identificados por uma URI, usados em documentos XML como tipos e nomes de atributos [W3C99]).

3.2.4 WSDL

WSDL (*Web Services Description Language* [W3C03a]) é uma linguagem baseada em XML usada para descrever Serviços Web e a forma como acessá-los. Ela especifica a localização do serviço e as operações (ou métodos) que o serviço expõe, permitindo uma separação da descrição da funcionalidade abstrata oferecida por um serviço dos detalhes de sua implementação.

WSDL descreve os Serviços Web a partir das *mensagens* que são trocadas entre o provedor de serviços e o requisitante:

- As mensagem são descritas de forma abstrata e então ligadas a um protocolo de rede.
- Uma mensagem consiste de uma coleção de ítens de dados com tipo definido.
- Uma troca de mensagens entre um provedor de serviços e seu requisitante é descrita como uma *operação*.
- Uma coleção de operações é chamada de *interface*.
- Uma interface é acessível através de um ou mais *endpoints*, tendo cada *endpoint* sua própria URI.

- Um serviço é uma coleção de *endpoints* ligados à mesma interface.

Um documento WSDL define um Serviço Web usando os seguintes elementos principais (Figura 3.7):

- <*portType*> - Operações realizadas pelo Serviço Web.
- <*message*> - As mensagens enviadas/recebidas pelo Serviço Web.
- <*types*> - Tipos de dados que o Serviço Web utiliza.
- <*binding*> - Protocolos de comunicação utilizados pelo Serviço Web.

```
<definitions>
    <types>
        definições de tipos........
    </types>
    <message>
        definições de mensagens....
    </message>
    <portType>
        definições de portas.......
    </portType>
    <binding>
        definições de binding....
    </binding>
</definitions>
```

Figura 3.7: Estrutura principal de um documento WSDL

O documento WSDL também pode conter outros elementos, como extensões e também um elemento do tipo *service* que torna possível agrupar em um único documento WSDL definições de diversos Serviços Web.

Um exemplo de documento WSDL é mostrado na Figura 3.8. Neste exemplo, o elemento *portType* define 'TermosDoGlossario' como o nome de uma porta, e 'obterTermo' como nome de uma operação. A operação 'obterTermo' possui uma mensagem de entrada (*input*) chamada 'obterTermoPedido' e uma mensagem de saída (*output*) chamada 'obterTermoResposta'. O elemento *message* define os componentes de cada mensagem (*part*) e os tipos de dados associados. Comparado a linguagens de programação tradicionais, 'TermosDoGlossario' seria uma biblioteca de funções e 'obterTermo' uma função tendo 'obterTermoPedido' como parâmetro de entrada e 'obterTermoResposta' como parâmetro de retorno.

```
<message name="obterTermoPedido">
    <part name="termo" type="xs:string"/>
</message>
<message name="obterTermoResposta">
    <part name="valor" type="xs:string"/>
</message>
<portType name="TermosDoGlossario">
    <operation name="obterTermo">
        <input message="obterTermoPedido"/>
        <output message="obterTermoResposta"/>
    </operation>
</portType>
```

Figura 3.8: Exemplo de parte de um documento WSDL

Tipos de Operações

WSDL define quatro tipos de operações, duas delas usadas por um serviço atuando como "cliente" e outras duas por um serviço atuando como "servidor".

1. *Operações do tipo "Cliente"*
   (a) *Solicit-response* (solicita-resposta): a operação envia um pedido e irá esperar por uma resposta.
   (b) *Notification* (notificação): a operação envia uma mensagem mas não espera resposta.
2. *Operações do tipo "Servidor"*
   (a) *One-way* (unidirecional): a operação pode receber uma mensagem mas não irá enviar uma resposta.
   (b) *Request-response* (pedido-resposta): a operação ao receber um pedido irá retornar uma resposta.

3.2.5 UDDI

UDDI (*Universal Description, Discovery and Integration*) é um *framework* independente de plataforma para descrever, descobrir e integrar serviços de negócio através da Internet. UDDI é um esforço multi-lateral da indústria que tem o apoio dos maiores fornecedores de software e plataformas, como Dell, Fujitsu, HP, Hitachi, IBM, Intel, Microsoft, Oracle, SAP, e Sun. Mais de 220 companhias são membros da comunidade UDDI.

À primeira vista, pode parecer simples gerenciar o processo de encontrar Serviços Web: se um parceiro de negócios conhecido tem um servidor de comércio eletrônico conhecido, o que falta descobrir? O que ocorre, entretanto, é que quando deseja-se descobrir quais parceiros de negócio possuem quais serviços, o problema torna-se mais complicado. Uma opção é telefonar a cada um dos parceiros e tentar encontrar a pessoa certa para conversar. Outra abordagem para resolver esse problema seria usar um arquivo de descrição dos Serviços Web no site de cada companhia. Essa abordagem distribuída é potencialmente escalável, mas peca em não ter um mecanismo que garanta a consistência dos formatos de descrição dos serviços. A abordagem de UDDI é baseada em um registro distribuído de negócios e as descrições de seus serviços, em formato XML [OAS00].

Os registros UDDI representam um diretório de interfaces de Serviços Web descritas em WSDL, e são baseados em padrões do W3C e da *Internet Engineering Task Force* (IETF) tais como XML, HTTP, SOAP e DNS.

Do ponto de vista de negócios, a especificação UDDI, em conjunto com WSDL, ajuda resolver os seguintes problemas:

- Descobrir os negócios corretos dentre as centenas de milhares atualmente on-line.
- Definir como realizar transações uma vez que o negócio é encontrado.
- Alcançar novos clientes e aumentar o acesso aos clientes atuais.
- Expandir as ofertas e aumentar o alcance de mercado.
- Descrever serviços e processos de negócio em um ambiente único, aberto e com razoável segurança.

O componente central do projeto UDDI é o registro de negócio, um arquivo XML usado para descrever uma entidade de negócios e seus Serviços Web. Conceitualmente, a informação contida em um registro de negócio UDDI consiste de três componentes:

- *White Pages* (Páginas Brancas): inclui endereço, contato e identificadores conhecidos.
- *Yellow Pages* (Páginas Amarelas): inclui uma categorização do negócio baseada em taxonomias padronizadas.
- *Green Pages* (Páginas Verdes): informações técnicas sobre os serviços que são expostos pelo negócio.

O modelo central usado pelos registros UDDI é definido com um *XML Schema*. Este *Schema* define quatro tipos principais de informação: informação do negócio *(businessEntity)*, informação do serviço *(businessService)*, informação de *binding (bindingTemplate)* e informações sobre especificação de serviços [OAS00].

3.3 Orquestração e Coreografia

Os Serviços Web estão migrando de seu modelo inicial *Descrever, Publicar, Interagir* para uma nova fase, na qual interações robustas que fazem parte de processos de negócio são suportadas. Essa fase é caracterizada pelo conceito de composição ou agregação de Serviços Web.

Esta composição de diversos serviços pode ser feita através de recursos como *Orquestração* e *Coreografia*. A *Orquestração* define um fluxo de interações (como em um *workflow*) entre diversos Serviços Web em um dado processo de negócio. Já através da *Coreografia* é possível estabelecer protocolos que definem as trocas de mensagem válidas entre um conjunto de múltiplos Serviços Web.

A diferença entre estes conceitos é, algumas vezes, bem tênue. Segundo o Novo Dicionário Aurélio [HF86], orquestração é '*a arte de distribuir as diferentes partes de uma peça musical, destinada a um conjunto, entre os diversos instrumentos deste conjunto, em função de seus respectivos timbres*' - realizando um paralelo no contexto de SOC, a *peça musical* pode ser considerada como um *processo de negócio*, os *instrumentos* como os *serviços* e os *timbres* como as *funcionalidades* de cada serviço. Já coreografia, ainda segundo o Novo Dicionário Aurélio da Língua Portuguesa [HF86], é '*a arte de conceber e compor a seqüencia de movimentos, passos e gestos de um bailado e de fazer a respectiva notação*' - a '*seqüencia de movimentos passos e gestos*' nada mais é, no contexto de SOC, do que os *protocolos que definem as trocas de mensagens válidas* entre os serviços (os *bailarinos* que compõem o *bailado* representam os serviços).

Pode-se considerar que a Orquestração está mais relacionada com a seqüencia de interações entre os serviços (lógica de negócio e ordem de execução, como em um *workflow* tradicional), enquanto que a Coreografia está mais preocupada com a troca pública de mensagens entre os serviços [Pel03], ou seja, como cada serviço deve ser comportar de forma a colaborar de forma harmoniosa com os demais. Normalmente com Orquestração existe uma entidade que coordena globalmente a composição, enquanto que com coreografia não existe tal controle.

Ao compor Serviços Web, seja através de orquestração ou de coreografia, alguns aspectos precisam ser considerados:

- As interações podem ser realizadas em qualquer ordem?
- Quais regras governam a seqüência das interações?
- Existe alguma relação entre as mensagens recebidas e/ou enviadas?
- Existe um "início"e um "fim"em uma dada seqüência de interações?
- Pode uma dada seqüência ser parcialmente desfeita?

- É possível desenhar uma visão global de todas as trocas de mensagem/interações?

Duas especificações que modelam a composição de Serviços web, permitindo a resposta de algumas das questões anteriores, e que merecem destaque no contexto deste trabalho são BPEL4WS e WSCI.

3.3.1 BPEL4WS

BPEL4WS (*Business Process Execution Language for Web Services*), ou simplesmente BPEL, é uma especificação recentemente publicada pela IBM, Microsoft e BEA que modela o comportamento dos Serviços Web em um processo de negócio aplicando conceitos de Workflow [SIM+03b]. A especificação define uma linguagem baseada em XML que descreve a lógica de controle requerida para coordenar os Serviços Web participantes em um fluxo de um processo. Essa descrição pode então ser interpretada e executada por um motor de orquestração, que é controlado por um dos participantes. Este motor coordena as diversas atividades do processo, e cuida da compensação do sistema quando erros ocorrem. BPEL é essencialmente uma camada sobre WSDL.

BPEL provê suporte tanto para processos de negócio executáveis, que essencialmente modelam *workflows* privados (abordagem de Orquestração), quanto para processos de negócio abstratos (abordagem de Coreografia).

O suporte a composição suportado por BPEL é orientado a processo: cada composição BPEL é um processo de negócio ou um *workflow* que interage com um conjunto de Serviços Web para alcançar um determinado objetivo. Essas composições são então chamadas de *processos* e os serviços com os quais os *Web Services* interagem são chamados de *parceiros* [CKM+03].

Um processo, como qualquer outro Serviço Web, suporta um conjunto de interfaces WSDL que o permite trocar mensagens com seus parceiros. O processo interage com os parceiros invocando as operações que eles suportam e recebendo mensagens através das interfaces de seus serviços. A interação entre uma composição BPEL e seus parceiros é, na maioria das vezes, do tipo *peer-to-peer*, na qual cada parte invoca operações (ou envia mensagens para) a interface pública do outro [CKM+03].

O núcleo de uma composição de processos BPEL é a definição das trocas de mensagem que ocorrem entre os processos e cada um dos seus parceiros. Primeiramente, os parceiros são definidos em um processo BPEL declarando-se as interfaces WSDL através das quais as interações irão ocorrer. De maneira a suportar um acesso multi-protocolo ao serviço, WSDL separa as descrições abstratas do serviço (interfaces e mensagens) da especificação de como será realizada a comunicação. Somente as interfaces abstratas são usadas na definição dos parceiros, o que torna as composições em BPEL independentes de plataforma

e do tipo de transporte, ou seja, o mesmo processo BPEL pode ser acessado através de SOAP sobre HTTP, bem como através de protocolos como IIOP e JMS.

Uma vez definidos os processos parceiros, um conjunto de primitivas é usado para definir como as mensagens serão trocadas com cada parceiro. Uma mensagem é enviada a um parceiro usando a primitiva *invoke*; o processo pode esperar por uma operação ser invocada por algum cliente externo utilizando a primitiva *receive*; a resposta de uma operação do tipo *pedido-resposta* é enviada usando a primitiva *reply*.

Além destas, BPEL provê outras primitivas para realizar ações como sinalizar falhas, terminar a execução de processos e manipular dados. Essas primitivas podem então ser combinadas em algoritmos mais complexos usando atividades estruturadas fornecidas pela linguagem. É possível definir uma seqüência ordenada de passos *(sequence)*, ramificações condicionais *(switch)*, laços *(while)*, executar um de vários caminhos *(pick)* e também indicar que uma coleção de passos deve ser executada em paralelo *(flow)*. Dentro das atividades sendo executadas em paralelo, restrições na ordem podem ser estabelecidas definindo-se ligações de controle entre as atividades [CKM+03].

Correlação de Mensagens

Para manter o estado de uma interação, BPEL usa um mecanismo denominado *correlação de mensagens*. Os campos principais nas mensagens de dados são identificados e usados para correlacionar as mensagens recebidas de um parceiro em uma dada conversa. Por exemplo, em um sistema de preenchimento de pedidos, o número da nota pode ser usado para identificar a conversa entre o processo e um de seus parceiros.

Tratamento de Falhas e Compensação

BPEL provê suporte extensivo para o tratamento de erros, através da utilização de mecanismos de compensação e de tratadores de falha. Os tratadores de falha oferecem um modelo estruturado para lidar com erros que ocorrem dentro de um processo; o modelo é similar aos blocos '*try-catch*' de Java, resultando em uma simplificação significativa do processo de modelagem requerido para tratar falhas. Em BPEL, este mecanismo está fortemente ligado com a noção de compensação (técnica usada para desfazer os efeitos de ações que já tenham se completado - por exemplo o cancelamento de uma reserva de passagem aérea). O projetista do processo define as ações de compensação que devem ser executadas caso um erro ocorra no curso da execução do processo - os mecanismos de compensação são tipicamente invocados por um tratador de falhas.

Em um processo BPEL, a menor unidade em que uma falha pode ocorrer é chamada de *escopo*. Se uma falha ocorre em um escopo, todas as atividades são desabilitadas e a falha é tratada ou passada para o escopo superior. Escopos já concluídos que estejam encadeados

com um que falhou são então compensados. O modelo de compensação de BPEL é fortemente relacionado com os protocolos definidos pela especificação *WS-Transaction* [SIM02].

WS-Coordination [SIM03a] e *WS-Transaction* são duas especificações que tratam de uma coordenação confiável e transacional dos Serviços Web [CKM+03]. Elas podem ser usadas individualmente ou em combinação com BPEL. Cada uma destas especificações tem um propósito bem-definido no contexto da composição robusta de serviços:

- *BPEL* permite que um conjunto de Serviços Web seja composto em um novo Serviço Web usando construtores de modelagem de processos bem definidos.
- *WS-Coordination* é um *framework* geral que permite a implementação de tipos específicos de coordenação, onde a coordenação dos Serviços Web requer um contexto compartilhado.
- *WS-Transaction* define dois tipos particulares de coordenação para transações atômicas (de curta duração) e também para transações de negócio (de longa duração).

O uso combinado destas três especificações permite que o modelo de composição BPEL seja estendido e passe a contar com capacidades de coordenação distribuídas.

3.3.2 WSCI

WSCI (*Web Services Choreography Interface*) é uma especificação da SUN, SAP, BEA e INTALIO que define uma linguagem baseada em XML para colaboração de Serviços Web [SISM02]. Ela define a coreografia como um todo, descrevendo as mensagens trocadas entre os Serviços Web. A especificação suporta correlação de mensagens, regras de sequenciamento, tratamento de exceções, transações e colaboração dinâmica. Um aspecto importante de WSCI é que somente o comportamento visível é descrito - WSCI não trata da definição de processos executáveis como BPEL. Além disso, um documento WSCI descreve somente a participação de um parceiro na troca de mensagens (uma coreografia baseada em WSCI inclui um conjunto de documentos WSCI, um para cada parceiro). Diferentemente de BPEL, em WSCI não há um processo que gerencie globalmente as interações.

Capítulo 4

Modelo da Plataforma VM-Flow

A plataforma *VM-Flow* [SM03] propõe uma infra-estrutura para um *Centro de Negócios Virtual* (*Virtual Marketplace*) que possui como características principais:

- Suporte às *Empresas Virtuais Dinâmicas*;
- Gerenciamento dos processos de negócio baseado em sistemas de *workflow*;
- Controle de execução inerentemente descentralizado;
- Proposição de uma série de Políticas de Interação;
- Interações interorganizacionais baseadas em mecanismos de Composição de Serviços (Orquestração e Coreografia).

Este capítulo apresenta o modelo da infra-estrutura e suas entidades (Seção 4.1), discute o suporte às EVDs (Seção 4.2), introduz as políticas de interação entre o VM-Flow e os parceiros membros das EVDs (Subseção 4.2.2). Na Seção 4.3 são apresentadas as Classes e Interfaces que compõem o Núcleo da Plataforma, na Seção 4.4 é introduzido um Exemplo de Aplicação construída sobre o *VM-Flow* e finalmente na Seção 4.5 outros Cenários de Uso são sugeridos.

4.1 Infra-estrutura

A infra-estrutura da plataforma *VM-Flow* é composta por um conjunto de *Facilidades* e por uma série de *Serviços de Suporte*, responsáveis por tarefas específicas e necessárias ao gerenciamento das EVDs e de suas interações. Um esquema geral da infra-estrutura do *VM-Flow* é apresentado na Figura 4.1.

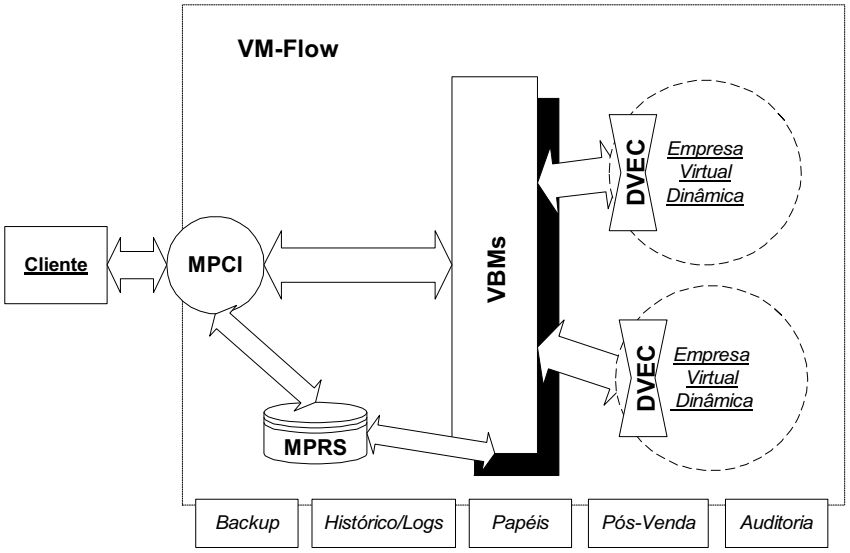


Figura 4.1: *VM-Flow*: Esquema geral da infra-estrutura.

4.1.1 Facilidades

As *Facilidades* são as entidades responsáveis pelo funcionamento básico do *VM-Flow* e das EVDs. São elas:

- MPCI (*Marketplace Customer Interface*): é a interface entre o VM-Flow e os clientes que desejam adquirir algum produto/serviço.
- MPRS (*Marketplace Repository Set*): conjunto de repositórios de dados e de serviços de acesso a tais dados.
- VBM (*Virtual Business Manager*): os Gerentes de Negócios Virtuais são os coordenadores dos processos de negócio em execução no *VM-Flow*.
- DVEC (*Dynamic Virtual Enterprise Coordinator*): responsável pelo gerenciamento do ciclo de vida das Empresas Virtuais Dinâmicas.

MPCI

É através da MPCI que todo tipo de contato dos clientes com o *marketplace* e com as EVDs ocorre. A MPCI, a partir das solicitações dos clientes, contata as outras entidades internas à infra-estrutura de forma a obter o que o cliente deseja. Essa separação entre o ambiente externo e interno do *marketplace* através de MPCIs possui algumas vantagens:

- Torna o processo interno do *VM-Flow* transparente ao cliente;
- Aumenta o alcance do *marketplace*, pois diferentes MPCIs podem ser construídas para diferentes categorias de clientes, mas todas vinculadas a uma mesma instância do *VM-Flow*;
- MPCIs baseadas em diferentes tecnologias podem ser construídas, ampliando a compatibilidade do *VM-Flow* sem necessidade de alterações em seu núcleo (exemplos de tecnologias: Serviço Web, interface web padrão, Java RMI, CORBA etc).

Alguns exemplos de interações que podem ocorrer na MPCI são:

- Consulta a produtos e serviços;
- Solicitação de propostas para realização de um determinado processo de negócio (pedido de compra de um produto/serviço);
- Negociação e aprovação de propostas;
- Acompanhamento do andamento dos pedidos;
- Solicitação de alterações nos pedidos;
- Interações pós-venda (Pesquisas de Satisfação, Reclamações, Acionamento de Garantias Legais)

MPRS

O MPRS é responsável pelo armazenamento de diferentes conjuntos de dados necessários ao funcionamento da infra-estrutura e utilizados pelas outras entidades do *VM-Flow*. São exemplos de dados de responsabilidade do MPRS:

- Catálogos de produtos e serviços oferecidos pelas empresas parceiras do *VM-Flow*;
- Contratos;
- Meta-informações da infra-estrutura;
- Históricos de Operações e Cópias de Segurança;
- Informações para Auditoria dos Processos de Negócio.

VBM

Os VBMs são responsáveis por diversas tarefas relacionadas com a execução de um dado processo de negócio. São tarefas dos VBMs:

- Elaborar propostas comerciais aos clientes baseadas nas informações que fazem parte do MPRS e que foram fornecidas pelos candidatos a parceiros do *VM-Flow*;
- Preparar os planos de execução associados às instâncias dos processos de negócio (*casos*);
- Selecionar/Criar o DVEC responsável pela Empresa Virtual que irá implementar o *caso*;
- Gerenciar todo o ciclo de vida de um dado *caso*, interagindo com o DVEC que cuida da execução do mesmo.

É importante ressaltar que um objeto VBM pode gerenciar diversas instâncias de processos de negócio, mas dada uma instância, só há um VBM associado a ela.

Quanto à sua distribuição, os VBMs são agrupados em *Agências de VBMs*, o que aumenta o grau de escalabilidade da plataforma. Além disso, diferentes categorias de VBMs especializadas de acordo com necessidades específicas de setores e nichos de mercado são suportadas pelo *marketplace* sem necessidade de alterações em sua infra-estrutura.

DVEC

O DVEC é a entidade responsável por criar e gerenciar as EVDs durante todo o seu ciclo de vida. Cada EVD possui um (e somente um) DVEC associado à ela desde sua criação até sua extinção.

As tarefas principais dos DVECs são:

- Selecionar membros (empresas "reais") para a EVD;
- Gerenciar as entradas e saídas de membros de uma EVD;
- Gerenciar os contratos eletrônicos entre os membros das EVDs e o *VM-Flow*. Estes contratos definem as responsabilidades de cada um, a forma de divisão de ganhos e também de eventuais prejuízos;
- Aplicar o plano de execução preparado pelo VBM, garantindo que os diversos parceiros que compõem a EVD cumpram suas tarefas determinadas;
- Renegociar, quando necessário, alterações dinâmicas no plano de execução e nos contratos a ele associados.

4.1.2 Serviços de Suporte

O *VM-Flow* também oferece alguns serviços adicionais em sua infra-estrutura, denominados *Serviços de Suporte*. Estes serviços são representados pelos seguintes gerentes:

- Gerente de Backup
- Gerente de Histórico/Log de Atividades
- Gerente de Auditoria
- Gerente de Papéis
- Gerente de Relacionamento Pós-Venda

Gerente de Backup

O *Gerente de Backup* é responsável por realizar e manter cópias de segurança dos dados dos diversos hosts que compõem a infra-estrutura, de forma a garantir uma recuperação segura do sistema em caso de falhas.

Gerente de Histórico/Log de Atividades

Juntamente com o serviço de *backup*, o *Gerente de Histórico* compõe o sistema de recuperação de falhas do *VM-Flow*. O primeiro evita a perda de dados enquanto que o segundo torna possível o *rollback* de operações que, por qualquer razão, precisem ser desfeitas.

Gerente de Auditoria

O *Gerente de Auditoria* permite a realização de avaliações da eficiência e de integridade dos processos de negócio executados pelo *VM-Flow* e também pelas empresas que participam das EVDs. Auditorias podem ser muito úteis na melhoria dos processos tanto da própria infra-estrutura quanto dos parceiros da mesma. Além disso, podem servir como garantia aos clientes de que seus processos estão sendo executados da forma contratada.

Gerente de Papéis

O *Gerente de Papéis* é o responsável pela alocação de recursos (serviços ou pessoas) que serão responsáveis pela execução de uma determinada tarefa do plano de execução. Este gerente é fundamental para o modelo de *workflow* em que o *VM-Flow* é baseado.

Gerente de Relacionamento Pós-Venda

A responsabilidade de contatar os clientes de forma a levantar seu grau de satisfação com relação aos produtos/serviços adquiridos é do *Gerente de Relacionamento Pós-Venda*. Ele também presta assistência com relação às garantias impostas por lei, acionando as empresas responsáveis quando necessário.

4.1.3 Gerenciamento Descentralizado da Plataforma

Na plataforma *VM-Flow* a instância de um processo de negócio (também chamada de *caso*) leva consigo seu próprio plano de execução enquanto migra entre diversos *hosts* para cumprir as diferentes tarefas e atividades do plano. Essa característica, inspirada na plataforma WONDER [SF00] (Seção 6.3), torna o controle dos processos de negócio inerentemente distribuído. A principal vantagem desta abordagem é a garantia de uma maior escalabilidade da plataforma, pois deixa de existir um "gargalo" no gerenciamento da mesma. Apesar disso, algumas desvantagens podem ser citadas, juntamente com as opções que adotamos para contorná-las:

- Perda do *caso* devido à falha no *host* em que ele se encontra em um dado momento: os *Gerentes de Suporte* apresentados anteriormente trabalham para contornar essa questão implementando um mecanismo eficiente de recuperação de falhas.
- Problemas com privacidade e integridade do Plano de Execução: as *Políticas de Interação* e o uso de Proxys nos membros das EVDs apresentados mais a frente neste capítulo são os responsáveis por resolver esta questão.

4.2 As Empresas Virtuais Dinâmicas

As funções de suporte às Empresas Virtuais Dinâmicas no *VM-Flow* são de responsabilidade do DVEC. Nesta seção apresentamos como é o ciclo de vida das EVDs e também as Políticas de Interação aplicadas junto aos parceiros membros destas EVDs.

4.2.1 Ciclo de Vida das EVDs

O ciclo de vida das EVDs no *VM-Flow* possui as seguintes etapas:

- *Formação*: ocorre na aprovação de um proposta por um cliente. O VBM cria um DVEC que então seleciona parceiros potenciais (já pré-inscritos no *VM-Flow*) para se tornarem membros de uma EVD. Neste momento contratos são estabelecidos entre o VM-Flow e os membros e também as Políticas de Interação (próxima subseção) são acordadas.

- *Manutenção*: o gerenciamento da EVD é realizado pelo DVEC e por seus *Proxies* (Subseção 4.2.4);
- *Alteração de Membros*: durante a vida de uma EVD, podem ocorrer, por diferentes razões, a inclusão e exclusão de membros, o que implica num re-arranjo do plano de execução por parte do VBM e do DVEC.
- *Extinção*: após a conclusão com sucesso do plano de execução, a EVD é mantida por algum tempo (para o caso de outro processo de negócio com as mesmas características ser executado) e depois é extinta. Todo o histórico de sua vida fica registrado.

4.2.2 Políticas de Interação

Para garantir um nível maior de flexibilidade, autonomia, privacidade e suportar diferentes tipos de colaboração entre os membros de uma EVD e o VM-Flow, nosso modelo estabelece um relacionamento inter-organizacional baseado em *Políticas de Interação*, estas classificadas em duas categorias distintas:

- *Políticas de Autonomia dos Parceiros*: define o tipo de relacionamento entre o *VM-Flow* e cada parceiro membro de uma EVD.
- *Políticas de Cooperação entre Parceiros*: define o tipo de relacionamento entre um dado par de parceiros dentro de uma EVD.

Políticas de Autonomia dos Parceiros

Esta categoria de *Políticas de Interação* estabelece o nível de interação entre o DVEC e um determinado membro de uma EVD. No momento em que uma empresa (real) se candidata a participar de uma EVD, uma negociação ocorre para definir qual o tipo de interação que ela deseja manter com o VM-Flow. O DVEC passa então a atuar (com esse parceiro específico) de uma das seguintes maneiras:

- *Supervisor*: interação através de uma interface bem definida com o *workflow* privado da empresa. O DVEC não tem acesso ao domínio interno do parceiro. O *DVEC Supervisor* pode ter os seguintes tipos de atuação:
  - Consultiva: o DVEC pode somente solicitar informações de estado sobre a instância de um processo.
  - Seletiva: o DVEC e o parceiro negociam em quais pontos do plano de execução interações serão permitidas.

  - Participativa: o DVEC pode interagir com todas as atividades que compõem o plano de execução (iniciar, pausar, reiniciar, cancelar, enviar/receber dados, checar status).

- *Coordenador*: o DVEC (através de um *Proxy* apresentado mais à frente) possui controle total sobre as tarefas do plano sendo executadas pelo *workflow* interno do parceiro, que torna-se uma extensão do *VM-Flow*.

O DVEC é o responsável por determinar as diferentes combinações de políticas de autonomia dentro de uma mesma EVD, baseado nas necessidades impostas pelo processo de negócio, pelo plano de execução e também pelas condições previamente negociadas com cada um dos parceiros.

Políticas de Cooperação entre Parceiros

Na interação entre os parceiros (empresas reais associadas em uma EVD), algumas das principais preocupações são como tratar a privacidade e a integridade dos dados que acompanham uma instância de um processo de negócio. Dentro deste contexto, a plataforma *VM-Flow* apresenta três *Políticas de Cooperação* diferentes entre dois parceiros membros de uma mesma EVD:

- *Cooperação Total*: os dois parceiros confiam totalmente um no outro. Quando um *caso* deixa um parceiro e se move para outro, não é necessário esconder nem o plano nem os dados do estágio anterior (essa informação muitas vezes é até necessária).

- *Cooperação Controlada*: existe um conjunto pré-estabelecido de informações que devem ser passadas ao próximo parceiro e um outro conjunto que deve ser escondido pelo DVEC (na verdade por seu *Proxy*).

- *Privacidade Total*: não existe nenhum compartilhamento de informação entre os parceiros. Todo o tipo de informação é retornado ao DVEC, que tem acesso ao plano e então decide o que fazer a seguir, escondendo do parceiro seguinte as atividades e dados do estágio anterior.

4.2.3 Políticas e Aplicações

As duas categorias de políticas de interação apresentadas (*Autonomia dos Parceiros* e *Cooperação entre Parceiros*) são ortogonais, ou seja, independentes. A seleção e combinação delas na formação de uma EVD dependem tanto de questões políticas (confidencialidade dos dados, por exemplo) quanto de limitações tecnológicas (grau de compatibilidade entre os sistemas das empresas, por exemplo). Algumas aplicações possíveis:

- *Provedor de Serviços de e-Business*: uma terceira entidade, independente, que oferece sua infra-estrutura e serviços a outras companhias que desejam participar de um *marketplace* virtual. Neste cenário, poderíamos associar uma Política de Autonomia do tipo *DVEC Supervisor* a uma Política de *Cooperação Controlada*.

- *Cadeia produtiva de uma Indústria Automobilística*: a montadora seria, por exemplo, a entidade detentora do *marketplace* e que estaria no controle dos processos de fabricação de seus fornecedores. Neste caso, o uso de uma Política de Autonomia do tipo *DVEC Coordenador* combinada com uma Política de Cooperação do tipo *Privacidade Total* seria uma alternativa interessante.

4.2.4 Interação entre o DVEC e a EVD

Com o objetivo de apresentar como é realizada a orquestração e a coreografia das atividades que compõem o plano de execução de um processo de negócio, é preciso analisar com mais detalhes a interação entre o DVEC e a EVD.

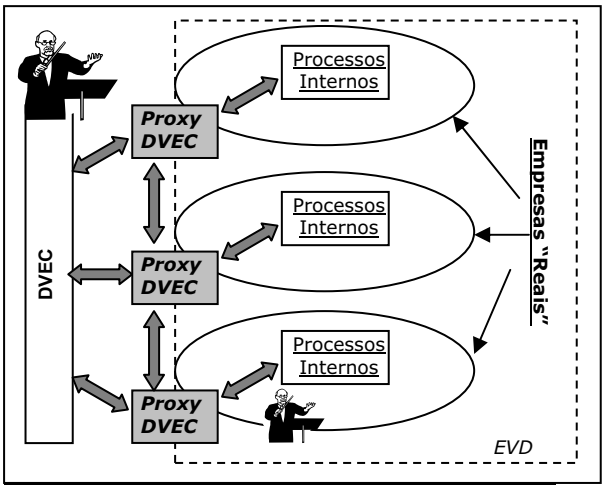


Figura 4.2: DVEC, DVE e Proxies

Um novo elemento é introduzido na Figura 4.2: o *Proxy DVEC*. Ele é o responsável por implementar a interação entre o *VM-Flow* e um determinado parceiro, executando a parte local do plano, sempre respeitando a Política de Autonomia acordada com o mesmo. Este *proxy* deve também "conversar", quando necessário, com o próximo parceiro (seguindo o plano de execução e o tipo de Política de Cooperação).

O DVEC torna-se então o responsável por Orquestrar o processo de forma global através dos diversos *Proxies*, sendo cada um destes o responsável pela orquestração/coreografia

da parte local do plano de execução (relacionado com um membro da EVD).

Algumas considerações adicionais relativas aos *Proxies*:

- Existe um *Proxy* associado a cada membro de uma empresa virtual (ou seja, a cada empresa *real*), criado no momento do ingresso na empresa virtual. Entretanto, como o *Proxy* está também associado a um único DVEC (e por conseqüência a uma única empresa virtual), se uma empresa *real* participar de diversas empresas virtuais, ela possuirá um *Proxy* viculado a cada uma destas associações. Isto é necessário para permitir que uma mesma empresa *real* tenha diferentes Políticas de Interação em cada uma destas Empresas Virtuais.

- Fisicamente o objeto *Proxy* se localiza na interface entre a empresa *real* e o mundo externo (no mesmo local de um servidor Web, por exemplo), normalmente sem acesso direto ao domínio interno da empresa.

- Como os *proxies*, juntamente com o DVEC, são os responsáveis diretos por colocar em prática as Políticas de Interação, *proxies* de diferentes categorias (apresentadas na Seção 4.3) podem co-existir dentro de uma mesma empresa virtual. Uma análise comparativa destas diferentes combinações é apresentada na Seção 4.2.5.

4.2.5 Análise Comparativa das Políticas de Interação

Uma das principais contribuições deste trabalho está relacionada com as diferentes *Políticas de Interação* propostas na Seção 4.2.2. A série de tabelas a seguir apresenta uma análise comparativa de diversos aspectos relacionados com estas políticas.

A Tabela 4.1 apresenta um resumo das combinações possíveis entre diferentes políticas. Já a Tabela 4.2 mostra alguns exemplos de cenários de uso com aplicação destas políticas.

Conforme já fora mencionado anteriormente, o principal responsável pela implantação das políticas junto aos membros das empresas virtuais é o objeto *Proxy DVEC* (Seção 4.2.4). A Tabela 4.3 apresenta de forma resumida como se comporta o *Proxy* com relação à execução da parte local do plano (tipo de composição), de acordo com a Política de Atuação em vigor.

Tabela 4.1: Resumo das Políticas de Interação

Política	**Atuação**	**Cooperação**
Sup. Consultiva + Privacidade Total	Pode somente solicitar informações de estado sobre um processo.	Não existe nenhum compartilhamento de informações entre os parceiros.
Sup. Consultiva + Cooper. Controlada	Pode somente solicitar informações de estado sobre um processo.	Um conjunto pré-acordado de informações é enviado ao próximo parceiro.
Sup. Consultiva + Cooper. Total	Pode somente solicitar informações de estado sobre um processo.	O caso e todos os seus dados são enviados diretamente ao próximo parceiro.
Sup. Seletiva + Privacidade Total	O DVEC e o parceiro negociam em quais pontos do plano de execução podem ocorrer interações.	Não existe nenhum compartilhamento de informações entre os parceiros.
Sup. Seletiva + Cooper. Controlada	O DVEC e o parceiro negociam em quais pontos do plano de execução podem ocorrer interações.	Um conjunto pré-acordado de informações é enviado ao próximo parceiro.
Sup. Seletiva + Cooper. Total	O DVEC e o parceiro negociam em quais pontos do plano de execução podem ocorrer interações.	O caso e todos os seus dados são enviados diretamente ao próximo parceiro.
Sup. Participativa + Privacidade Total	O DVEC pode interagir com todas as atividades que compõem o plano de execução.	Não existe nenhum compartilhamento de informações entre os parceiros.
Sup. Participativa + Cooper. Controlada	O DVEC pode interagir com todas as atividades que compõem o plano de execução.	Um conjunto pré-acordado de informações é enviado ao próximo parceiro.
Sup. Participativa + Cooper. Total	O DVEC pode interagir com todas as atividades que compõem o plano de execução.	O caso e todos os seus dados são enviados diretamente ao próximo parceiro.
Coordenadora + Privacidade Total	O DVEC pode interagir com todas as atividades que compõem o plano de execução.	Não existe nenhum compartilhamento de informações entre os parceiros.
Coordenadora + Cooper. Controlada	O DVEC pode interagir com todas as atividades que compõem o plano de execução.	Um conjunto pré-acordado de informações é enviado ao próximo parceiro.
Coordenadora + Cooper. Total	O DVEC pode interagir com todas as atividades que compõem o plano de execução.	O caso e todos os seus dados são enviados diretamente ao próximo parceiro.

Tabela 4.2: Políticas e Aplicações

Política	**Cenário de Aplicação**
Sup. Consultiva + Privacidade Total	Provedor de Serviços de e-Business com participante que deseja oferecer produto/serviço de forma 100% autônoma.
Sup. Consultiva + Cooper. Total	Turismo (hotel x companhia aérea x locadora de carro)
Sup. Seletiva + Privacidade Total	Parceiros Concorrentes em Potencial
Sup. Seletiva + Cooper. Controlada	Provedor de Serviços de e-Business
Sup. Seletiva + Cooper. Total	Mercado de Construção Civil
Sup. Participativa + Privacidade Total	Licitações Governamentais
Sup. Participativa + Cooper. Controlada	Indústria de Informática
Coordenadora + Privacidade Total	Indústria Automobilística e seus fornecedores.
Coordenador + Cooper. Total	Empresa controladora de grupo

Tabela 4.3: Políticas de Atuação e Composição de Serviços

Política	**Composição Realizada no Proxy**
Supervisora Consultiva	Coreografia (opcional) e consultas de estado.
Supervisora Seletiva	Orquestração parcial das atividades internas.
Supervisora Participativa	Orquestração total das atividades internas.
Coordenadora	O *Proxy* acessa diretamente o *workflow* interno e seus recursos.

4.3 Classes e Interfaces do Núcleo da Plataforma

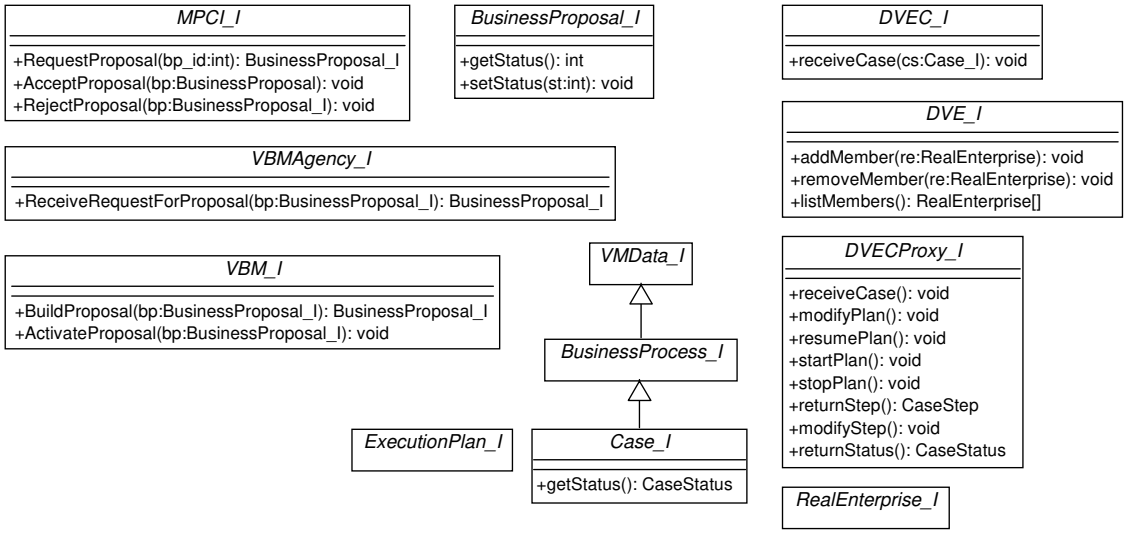


Figura 4.3: Diagrama de Interfaces do Núcleo do VM-Flow

As interfaces das principais classes que compõem o núcleo da plataforma, juntamente com seus métodos mais representativos, são apresentadas na Figura 4.3 e descritas a seguir[1]:

- *MPCI_I*: representa a entidade MPCI do ponto de vista interno ao *VM-Flow*. Ela exporta uma interface WSDL para comunicação com o mundo externo.
  - `requestProposal()`: método chamado por um cliente, através de um Serviço Web, solicitando uma proposta comercial para algum produto/serviço.
  - `acceptProposal()`: método chamado por um cliente, através de um Serviço Web, que indica a aceitação de uma proposta comercial.
  - `rejectProposal()`: método chamado por um cliente, através de um Serviço Web, que indica a rejeição de uma proposta comercial.
- *VBMAgency_I*: agrupa objetos que implementam *VBM_I*.
  - `receiveRequestForProposal()`: chamado pela MPCI indicando que será necessária a elaboração de uma proposta comercial. A agência VBM, baseada no tipo de produto/serviço, seleciona um VBM apropriado e encaminha a solicitação para ele.

[1] Os parâmetros e tipos de retorno dos métodos foram suprimidos da descrição a seguir mas constam da Figura 4.3.

- *VBM_I*: base para a especialização dos VBMs.
  - `buildProposal()`: método chamado pela agência VBM solicitando a elaboração de uma proposta comercial.
  - `activateProposal()`: após aceitação do cliente, a MPCI chama este método no VBM indicando que a proposta deverá ser colocada em prática.
- *DVE_I*: representa a Empresa Virtual Dinâmica. Uma classe que implementa *DVE_I* agrupa objetos que implementam *RealEnterprise_I*.
  - `addMember()`: o DVEC chama este método quando deseja adicionar um novo membro à Empresa Virtual Dinâmica (DVE).
  - `removeMember()`: usado pelo DVEC para remover um membro de uma Empresa Virtual.
  - `listMembers()`: lista os membros participantes na Empresa Virtual.
- *RealEnterprise_I*: empresa "real", membro de uma EVD.
- *DVEC_I*: base para criação de DVECs.
  - `receiveCase()`: método usado para transferir o *caso* (instância do processo de negócio sendo executado) para o DVEC.
- *DVECProxy_I*: base para a família de *Proxies* DVEC.
  - `receiveCase()`: utilizado na transferência do *caso*.
  - `modifyPlan()`: utilizado na alteração da parte local do plano de execução.
  - `startPlan()`: inicia a execução local do plano.
  - `stopPlan()`: para/pausa a execução local do plano.
  - `resumePlan()`: reinicia uma execução anteriormente pausada do plano.
  - `returnStep()`: devolver informações sobre a etapa do plano sendo atualmente executada.
  - `modifyStep()`: modifica uma determinada etapa do plano.
  - `returnStatus()`: retorna informações de estado do objeto *proxy* e também do *caso*.
- *BusinessProposal_I*: proposta de negócio que, caso aprovada, torna-se a base para a construção do plano de execução e do *caso*.

  - `getStatus()`: informa o estado de uma proposta de negócio (em elaboração, em avaliação, em negociação, aprovada, reprovada).
  - `setStatus()`: altera o estado de uma proposta de negócio.

- *ExecutionPlan_I*: plano de execução.
- *VMData_I, BusinessProcess_I, Case_I*: todos os dados trocados entre os objetos que compõem o *VM-Flow* herdam de *VMData_I*. *BusinessProcess_I* é usado para criar os *templates* dos processos de negócio, enquanto que *Case_I* irá representar as instâncias destes processos.

Um determinado processo de negócio, ao ser instanciado, se torna o que chamamos de *caso*, representado pela classe *Case* (Figura 4.4). Os *casos* migram entre as facilidade (Subseção 4.1.1) e os parceiros do *VM-Flow* levando consigo o plano de execução (veja que *ExecutionPlan* é um dos atributos de *Case*), o que implementa o controle descentralizado apresentado na Seção 4.1.3. Esta migração é efetivada através do método *ReceiveCase()*, presente em várias das classes apresentadas na Figura 4.4.

Outra característica importante ilustrada nesta figura é a hierarquia das classes da família *DVECProxy*: *DVECProxy_SP*, *DVECProxy_SC*, *DVECProxy_SS* e *DVECProxy_CC* implementam as diferentes Políticas de Autonomia junto aos parceiros do *VM-Flow*. Estas classes são as responsáveis por coordenar a execução da parte local do plano, de acordo com as políticas selecionadas, e também por realizar a orquestração/coreografia dos serviços internos aos membros das EVDs (vale lembrar que existe um objeto descendente de *DVECProxy* associado a cada membro de uma EVD).

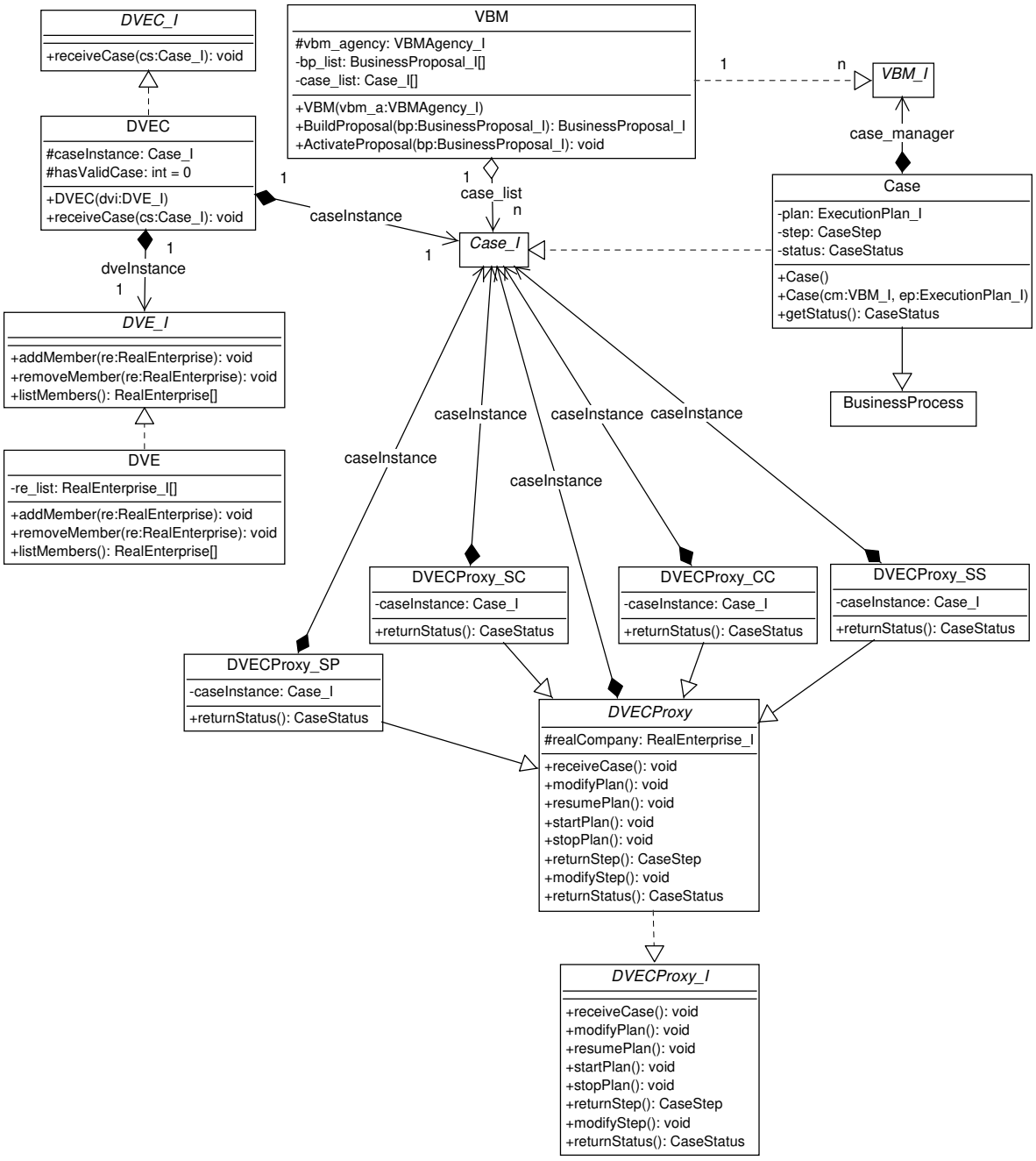


Figura 4.4: Classes do Núcleo da Plataforma

4.4 Exemplo de Aplicação

A aplicação escolhida para validar nossa infra-estrutura é a de uma indústria de computadores hipotética chamada LEED. No modelo de negócio adotado, esta indústria realiza apenas a montagem (integração) dos diversos componentes de um computador, não sendo responsável pela fabricação destes. Dessa forma, a LEED utiliza a plataforma *VM-Flow* para encontrar fornecedores de componentes que irão atender às necessidades de seus clientes.

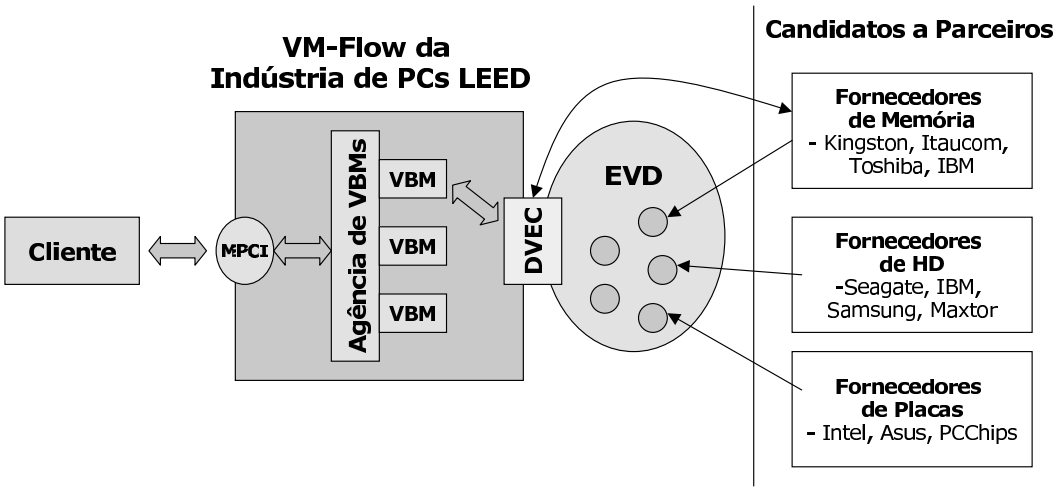


Figura 4.5: Exemplo de Aplicação: Indústria de PCs LEED

Na Figura 4.5, um esquema geral do *VM-Flow* usado pela LEED é apresentado, com as seguintes peculiaridades:

- O *Cliente* pode ser um revendedor de equipamentos ou um ainda um comprador corporativo de grande porte;
- Três tipos de VBMs são definidos com o objetivo de atender às diferentes categorias de produtos oferecidos pela LEED:
  - VBM para Desktops;
  - VBM para Notebooks;
  - VBM para Servidores.
- Os *Candidatos a Parceiros* representam os fornecedores em potencial dos componentes necessários à montagem dos produtos oferecidos pela LEED.

Um processo de negócio típico da LEED consiste das seguintes etapas (Figura 4.6):

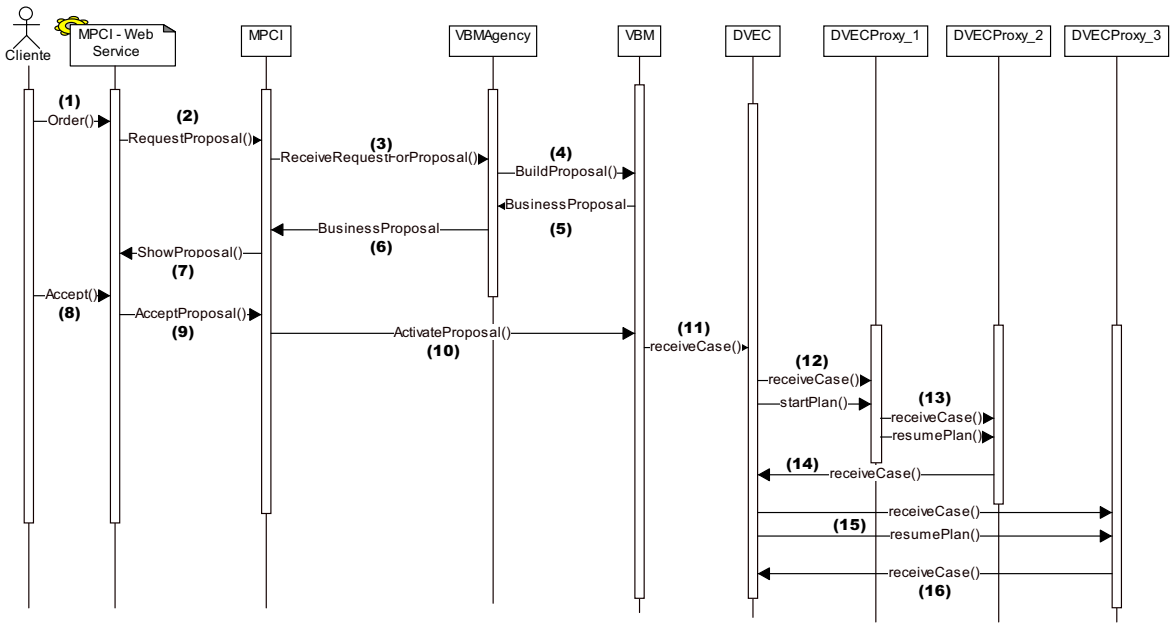


Figura 4.6: Processo de Negócio da LEED

- O Cliente interage com a MPCI, consultando as informações referentes aos produtos oferecidos e solicitando uma proposta comercial (1,2).
- A MPCI contata a Agência de VBMs (3). De acordo com o tipo de produto solicitado, o VBM adequado é alocado para cuidar da proposta comercial (4).
- O VBM, baseado nas necessidades do cliente e nas informações oferecidas pelos fornecedores potenciais, elabora uma proposta comercial e um pré-plano de execução (5,6). A proposta é apresentada ao cliente através da MPCI (7).
- No caso de aprovação da proposta (8,9,10), o VBM imediatamente cria um DVEC, passando a ele um pré-plano de execução (11). Esse DVEC seleciona então os membros da EVD e finaliza com o VBM a definição do plano de execução.
- A partir deste instante, o DVEC torna-se o responsável direto pela execução do plano, interagindo com os *Proxies* localizados em cada um dos parceiros membros da EVD (12). Note que o membro associado ao *DVECProxy_1* possui uma relação do tipo Cooperativa com o membro associado ao *DVECProxy_2*, pois ele passa o *caso* diretamente ao seu parceiro (13). O mesmo não ocorre na relação parceiro 2 x parceiro 3 - o *DVECProxy_2* é obrigado a devolver o *caso* para o DVEC, o qual encaminha o mesmo ao *DVECProxy_3* (já com as devidas restrições de privacidade

de informação impostas pela política entre 2 e 3) (14,15,16).

4.5 Outros Cenários de Uso

A seguir são apresentados outros exemplos de cenários nos quais a plataforma VM-Flow pode ser aplicada (além dos já citados na Subseção 4.2.3 - *Provedor de Serviços de e-Business* e *Indústria Automobilística*).

Mercado de Turismo

Em um cenário clássico em Empresas Virtuais, o cliente (ou uma Agência de Turismo representando um cliente), consulta o *VM-Flow* para encontrar hotéis, companhias aéreas e locadoras de automóveis no planejamento de uma viagem.

Em outro exemplo, uma excursão que será oferecida a diversos grupos de turistas é montada por uma Agência/Operadora, associada a uma EVD. Neste contexto poderiam ser aplicadas:

- Relação hotel x companhia aérea x locadora de carro: políticas de autonomia do tipo *DVEC Supervisor Consultivo* e de *Cooperação Total*;
- Relação entre empresas potencialmente concorrentes (duas companhias aéreas que voam o mesmo trecho, por exemplo): política de cooperação do tipo *Privacidade Total*.

Construção Civil

Imobiliárias, fornecedores de material, empreiteiras, escritórios de engenharia e arquitetura podem se associar a um *VM-Flow* especializado para esse nicho de mercado e passar a oferecer serviços de projeto, construção e decoração de casas.

Este cenário parece adequado para um ambiente de *Cooperação Total* dentro de uma EVD, exceto para o caso de membros que são potenciais concorrentes (fornecedores de Material de Construção, por exemplo).

Licitações Governamentais

A plataforma poderia ser adaptada (através da especialização de VBMs) para oferecer suporte a licitações públicas. Sendo o Governo o detentor da plataforma, políticas de autonomia com *DVEC Supervisor Participativo* ou ainda *DVEC Coordenador* poderiam ser aplicadas neste cenário.

Capítulo 5

Implementação

Com o objetivo de comprovar a validade do modelo apresentado no Capítulo 4, realizamos a implementação de um protótipo da plataforma *VM-Flow* contemplando as principais funcionalidades da mesma.

Nossa escolha foi de realizar a implementação em uma linguagem orientada a objetos, mais especificamente Java (para alcançar uma maior independência de plataforma). O acesso à MPCI e aos sistemas internos dos membros das Empresas Virtuais é realizado via Serviços Web, sendo que a orquestração/coreografia é implementada através da especificação BPEL4WS. O motor BPEL usado foi o BPWS4J [IBM03]. Além do protótipo da infra-estrutura, uma aplicação-exemplo (*Indústria de PCs LEED* - Seção 4.4) construída sobre ela também foi implementada.

5.1 Protótipo da Plataforma

Facilidades

A Tabela 5.1 apresenta o *status* de implementação das diversas *Facilidades* que compõem o núcleo da plataforma *VM-Flow* e que foram introduzidas no capítulo anterior. O protótipo oferece suporte para que estas facilidades estejam localizadas tanto na mesma estação quanto distribuídas. Nesta última situação a comunicação entre os objetos Java é feita através de chamadas do tipo Java/RMI.

Tabela 5.1: Núcleo da Plataforma - Status de Implementação das Facilidades

Facilidade	**Implementada?**	**Observações**
MPCI	Sim	Objeto Java + Serviço Web
MPRS	Sim	Objeto Java + Base de Dados
VBM	Parcialmente[1]	Objeto Java
Agência VBM	Sim	Objeto Java
DVEC	Parcialmente[1]	Objeto Java + Motor de Orquestração
Proxies DVEC	Sim	Objeto Java + Motor de Orquestração + Serviço Web

(1) Os mecanismos de Gerência das EVDs foram implementados por completo. Aspectos relacionados com Negociação de Contratos não foram implementados.

Serviços de Suporte

A Tabela 5.2 apresenta o *status* de implementação dos *Serviços de Suporte* propostos no Capítulo 4.

Tabela 5.2: Serviços de Suporte - Status de Implementação

Serviço	**Implementado?**	**Observações**
Backup	Parcialmente (exceto Recuperação de Falhas)	Uma instância em cada estação participante, todas vinculadas a um Gerente principal. Armazenamento dos dados via MPRS.
Histórico/Log	Parcialmente (exceto *Rollback*)	Uma instância em cada estação participante, todas vinculadas a um Gerente principal. Armazenamento dos dados via MPRS.
Auditoria	Não	—
Papéis	Sim	Uma instância junto à cada DVEC/EVD.
Relacionamento Pós-Venda	Não	—

5.2 Composição dos Serviços

Nosso modelo de orquestração e coreografia de serviços é construído sobre a especificação BPEL4WS. BPEL define dois modelos de processos de negócio:

- *Processos de Negócio Executáveis*: modela o comportamento dos parceiros em um processo de negócio, da mesma forma que um *workflow* privado. Estas interações são executadas por um motor de orquestração.
- *Processos de Negócio Abstratos*: modelados como protocolos de negócio, especificam as trocas públicas de mensagens entre as partes. Os protocolos de negócio não são executáveis e não tratam dos detalhes internos do processo.

Essencialmente, os processos executáveis oferecem suporte à orquestração enquanto os protocolos de negócio têm seu foco na coreografia dos serviços.

5.2.1 Níveis de Composição de Serviços no VM-Flow

A composição dos serviços no *VM-Flow* ocorre em dois níveis:

1. O DVEC orquestra seus *Proxies*: cada *Proxy* é implementado como uma classe JAVA que tem sua interface exportada como um Serviço Web. O plano de execução global consiste da composição destes Serviços Web através de um *Processo de Negócio Executável* BPEL.
2. Cada *Proxy* é responsável pela orquestração (ou coreografia) do plano de execução local - de acordo com a Política de Autonomia acordada entre o membro da EVD e o *VM-Flow*, o mecanismo correspondente (orquestração ou coreografia) é adotado localmente.

Interação entre o VM-Flow e os Parceiros de Negócio

Quando o DVEC atua como Coordenador, o *VM-Flow*, através do *Proxy DVEC*, usa as definições de processo de negócio executável do BPEL para orquestrar o plano dentro do membro da EVD. Já quando o DVEC atua como Supervisor, as definições abstratas são utilizadas em um contexto de coreografia.

Interação entre Parceiros

Quando políticas de Cooperação (Total ou Controlada) são selecionadas, os processos de negócio abstratos são úteis na definição de protocolos de negócio usados nas mensagens trocadas entre os *proxies* (contexto de coreografia).

5.2.2 Composições BPEL

Conforme mencionado anteriormente, dois níveis de composição de serviços ocorrem no *VM-Flow*:

1. O DVEC coordena a composição dos serviços de seus *Proxies*.
2. Cada *Proxy* coordena a composição dos serviços do membro da EVD que ele está associado.

O tipo de composição realizada, a forma como essa composição é realizada e, por conseqüência, o recurso BPEL utilizado, são determinados pelas Políticas de Interação em vigor.

O mecanismo usado pelo DVEC para compor os serviços dos proxies é sempre Orquestração, através de um Processo de Negócio Executável BPEL. O que muda é a forma como essa orquestração é construída (o fluxo do *caso*, por exemplo). A Tabela 5.3 apresenta estes diferentes fluxos de acordo com a Política de Cooperação entre dois parceiros A e B quaisquer, onde pA e pB indicam os *proxies* correspondentes.

Tabela 5.3: Políticas de Cooperação e a Orquestração no DVEC

Política entre A e B	**Fluxo do Caso entre dois Proxies pA e pB**
Privacidade Total	DVEC » pA » DVEC » pB » DVEC
Cooperação Controlada	DVEC » pA » pB (+ DVEC) » DVEC
Cooperação Total	DVEC » pA » pB » DVEC

Legenda: O símbolo "»" indica "transfere *caso* para".

Já dentro de cada um dos *proxies*, podem ocorrer, de acordo com a Política de Autonomia em vigor, diferentes combinações de tipos de composição. A Tabela 5.4 apresenta de maneira resumida estas combinações, juntamente com o tipo de Processo BPEL envolvido.

Tabela 5.4: Políticas de Autonomia e Composição no Proxy

Política de Atuação	**Composição**	**Processo BPEL**
Supervisora Consultiva	Coreografia (opc.)	Abstrato
Supervisora Seletiva	Orquestração (parc.) + Coreografia (opc.)	Executável + Abstrato
Supervisora Participativa	Orquestração	Executável
Coordenadora	Não há. Acesso direto ao SGWF interno.	—

A seguir dois exemplos de Composições BPEL: uma orquestração executada por um DVEC e uma coreografia gerenciada por um *Proxy*.

Composição no DVEC

A Figura 5.1 apresenta um exemplo de Processo de Negócio Executável BPEL usado pelo DVEC para realizar a orquestração de seus *proxies*. O exemplo ilustra diversas migrações de um *caso* entre o DVEC e seus *proxies*, localizados nos membros das EVDs. Algumas observações:

- `<invoke>` apresenta uma chamada a uma operação localizada em um serviço web;
- `<receive>` prepara o processo BPEL para receber uma chamada externa de um serviço.
- O atributo `name` é usado para identificar uma atividade.
- O atributo `partnerLink` indica qual o parceiro de negócio, neste caso um dos *Proxies* DVEC, envolvido na atividade.
- O atributo `operation` determina qual é a operação relacionada com a atividade.
- O atributo `variable` é usado para identificar os dados (enviados ou recebidos) que fazem parte de uma atividade.
- O marcador `<sequence>` indica um bloco de atividades que devem ser executadas em série, enquanto que o marcador `<flow>` determina um bloco de atividades a serem executadas em pararelo;

As seguintes atividades compõem o processo apresentado no exemplo (Figura 5.1):

```
<sequence>
    <invoke name="sendCase_1" partnerLink="proxy_1" operation="receiveCase"
      inputVariable="caseInstance" />
    <receive name="receiveCase_1" partnerLink="proxy_1" operation="receiveCase"
      variable="caseInstance" /receive>
    <invoke name="sendCase_2" partnerLink="proxy_2" operation="receiveCase"
      inputVariable="caseInstance" />
    <receive name="returnWhenReady_6" partnerLink="proxy_6"
      operation="returnWhenReady" variable="fool" /receive>
    <receive name="returnWhenReady_7" partnerLink="proxy_7"
      operation="returnWhenReady" variable="fool2" /receive>
    <flow>
        <invoke name="resumePlan_6" partnerLink="proxy_6"
         operation="resumePlan"/>
        <invoke name="resumePlan_7" partnerLink="proxy_7"
         operation="resumePlan"/>
    </flow>
    <receive name="receiveCase_7" partnerLink="proxy_7" operation="receiveCase"
     variable="caseInstance" /receive>
</sequence>
```

Figura 5.1: Exemplo de Processo Executável BPEL em um DVEC

1. Inicialmente o DVEC (responsável pela execução da orquestração) realiza a transferência do *caso* para o primeiro parceiro de seu plano de execução, representado neste exemplo por `proxy_1`;

2. Em seguida, o DVEC aguarda uma chamada de *proxy_ 1* com o retorno do *caso*. O fato de *proxy_ 1* retornar o *caso* diretamente ao DVEC (e não ao próximo membro do plano de execução - proxy_2) indica que a Política de Cooperação entre proxy_1 e proxy_2 é do tipo Privacidade Total;

3. O DVEC envia o *caso* para o *proxy_ 2*;

4. A quarta e a quinta operação representam consultas de *Status* no *proxy_ 6* e no *proxy_ 7*. As operações `returnWhenReady` só terão retorno quando estes *proxies* chegarem em uma determinada etapa de seu plano de atividades local - essa informação será necessária para o próximo fluxo de atividades. Observe também que não há nenhuma operação do tipo `invoke` ou `receive` após o envio do *caso* para o *proxy_ 2*: isto indica que as relações *proxy_ 2 X proxy_ 3*, *proxy_ 3 X proxy_ 4*, *proxy_ 4 X proxy_ 5*, *proxy_ 5 X proxy_ 6* e *proxy_ 6 X proxy_ 7* são de Cooperação

Total, não sendo necessária a intervenção do DVEC nestas migrações do *caso*.

5. Após a confirmação de que *proxy_ 6* e *proxy_ 7* estão na etapa desejada e prontos para continuar, um fluxo (<flow>) de atividades em paralelo é executado. As operações dentro deste fluxo são do tipo resumePlan e solicitam o reinício da execução local em cada um dos *proxies* em paralelo.

6. Finalmente, o DVEC aguarda o retorno do *caso* a partir do *proxy_ 7* e então finaliza esta seqüencia de operações.

Composição nos Proxies

A Figura 5.2 apresenta um exemplo de Processo de Negócio Abstrato BPEL que define as trocas públicas de mensagem em uma coreografia composta por um *proxy* e pelos serviços oferecidos localmente por um membro de uma EVD. Sua estrutura é similar à de um processo executável - a diferença é que o motor de orquestração BPEL usa sua definição como um protocolo, somente para validar uma dada seqüência de mensagens trocadas por outros processos sendo executados, mas sem interferir no processo.

```
<sequence>
    <invoke name="task_1" partnerLink="service_1"
     operation="operation_1" />
    (...)
    <receive name="task_n" partnerLink="service_n"
     operation="operation_3" /receive>
</sequence>
```

Figura 5.2: Exemplo de Processo Abstrato BPEL em um Proxy

5.3 A Aplicação LEED

A aplicação *LEED* (Indústria de Computadores), apresentada na Seção 4.4, é construída sobre a plataforma *VM-Flow* e possui uma interface Web que interage com os serviços oferecidos pela MPCI. Neste exemplo, a *LEED* é a detentora também do *VM-Flow*.

Três níveis de usuário são definidos:

- *Cliente*: clientes (empresas/consumidores) que irão adquirir produtos da *LEED*.
- *Fornecedor*: todos os fornecedores em potencial que estão cadastrados no *VM-Flow* e classificados como parceiros da *LEED*.

- *Administrador LEED*: usuário interno da indústria. Essencialmente monitora e administra a plataforma e os processos de negócio da *LEED*.

As figuras a seguir ilustram o funcionamento da aplicação.

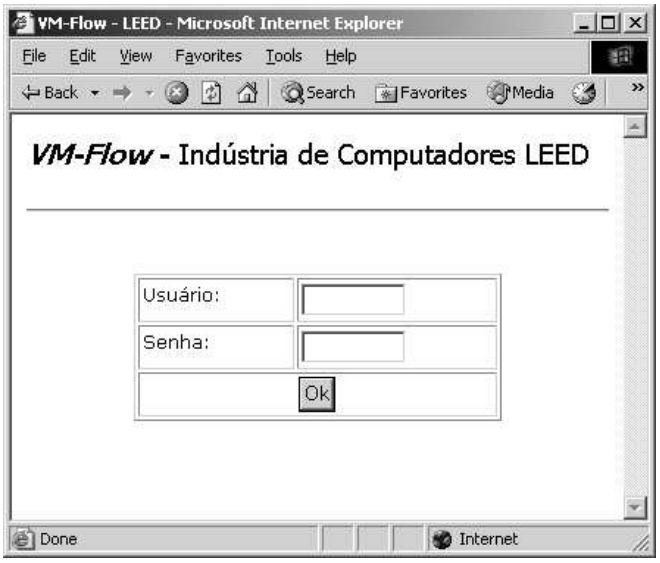


Figura 5.3: LEED - Tela Inicial

A Figura 5.3 apresenta a tela inicial (interface Web) da Aplicação LEED. Já a Figura 5.4 apresenta um formulário utilizado pelos clientes para Solicitação de Propostas. No exemplo em questão, o Cliente (*JJJEnterprises*) solicita uma proposta para a fabricação um servidor.

Após a solicitação da proposta pelo Cliente, o *VM-Flow* localiza fornecedores em potencial para cada um dos ítens. A Figura 5.5 mostra a visão de um destes fornecedores em potencial (*MMMemory* - Fabricante de Memórias). No exemplo ela possui três propostas de parcerias em potencial, sendo que uma delas é a solicitada pela *JJJEnterprises* na Figura 5.4. Depois da *MMMemory* e de todos os demais fornecedores negociarem com a LEED as condições da proposta (preços, prazos etc), a mesma é encaminhada à apreciação do Cliente (Figura 5.6). Neste momento o Cliente pode *Aprovar* a proposta, *Rejeitar* ou ainda tentar uma *Renegociação*.

A Figura 5.7 apresenta todas as parcerias em andamento (fruto de propostas aprovadas) do Fornecedor *MMMemory*. Uma destas parcerias consiste no fornecimento de memória para o servidor da *JJJEnterprises*. A última tela apresentada (Figura 5.8) apresenta a visão detalhada de um Administradror da LEED (detentora do *VM-Flow*) sobre um processo de negócio em execução. O exemplo mostra as diversas atividades necessárias ao fornecimento do servidor encomendado pela *JJJEnterprises*.

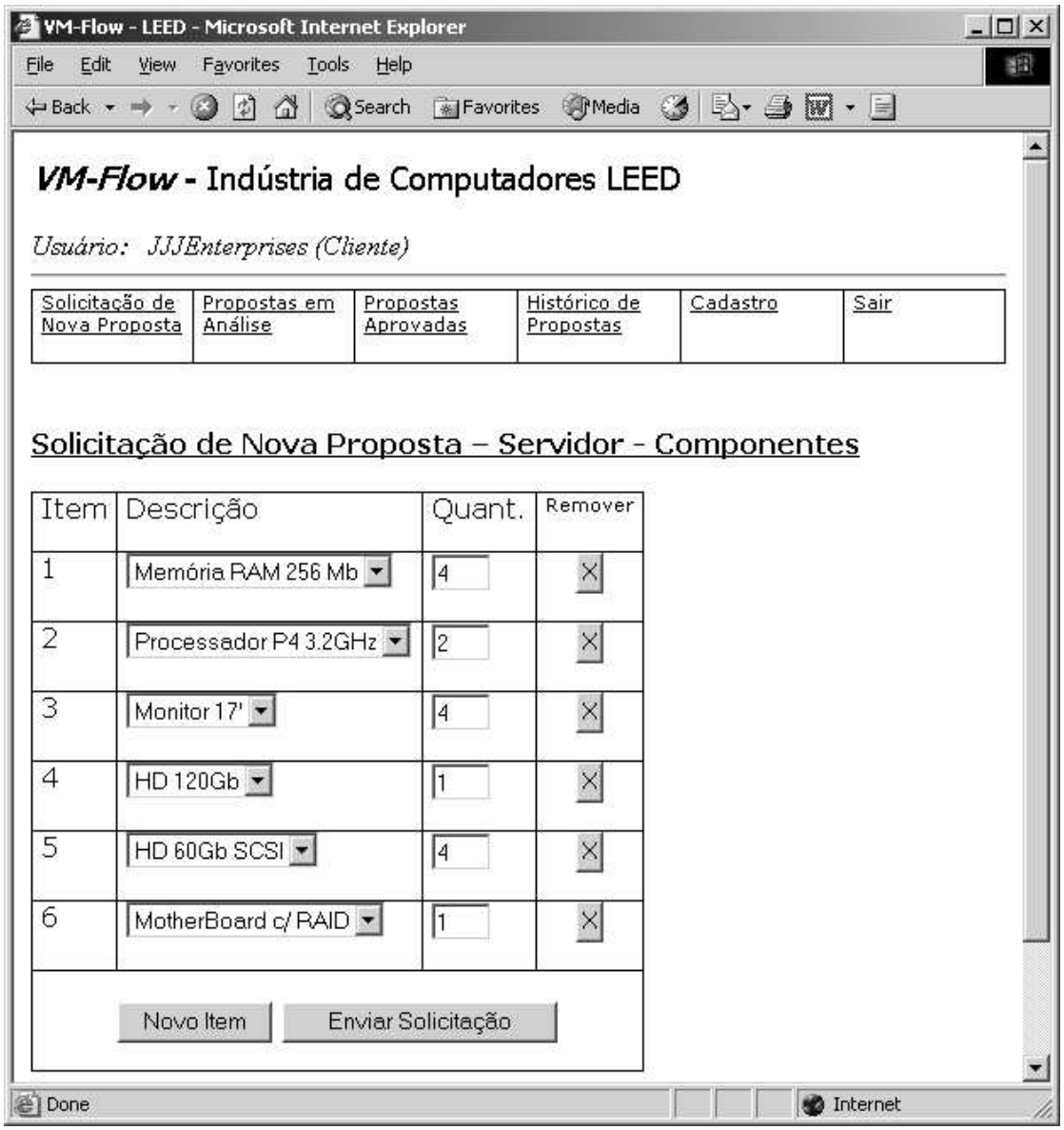


Figura 5.4: LEED - Cliente - Solicitação de Proposta

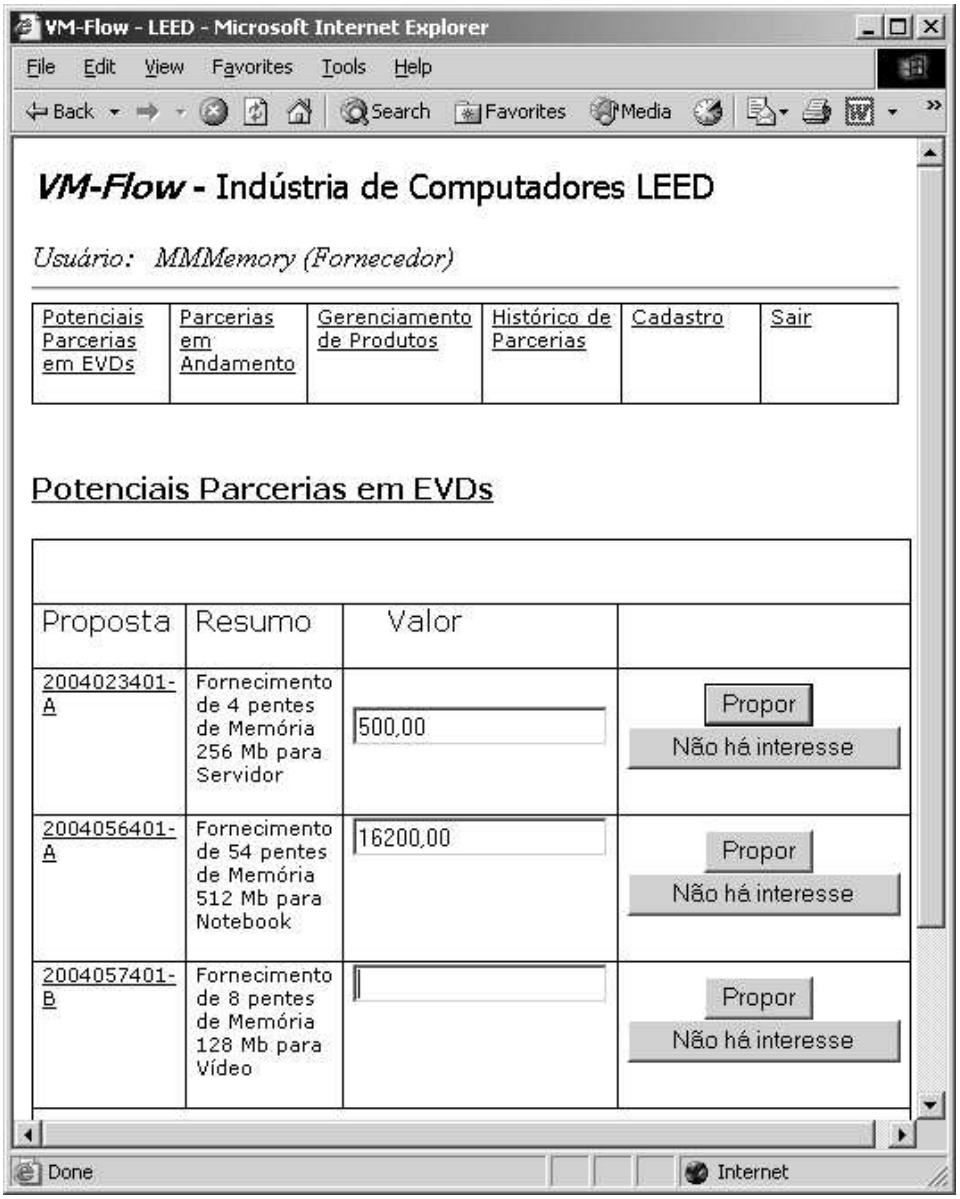


Figura 5.5: LEED - Fornecedor - Potenciais Parcerias

Figura 5.6: LEED - Cliente - Proposta de Negócio

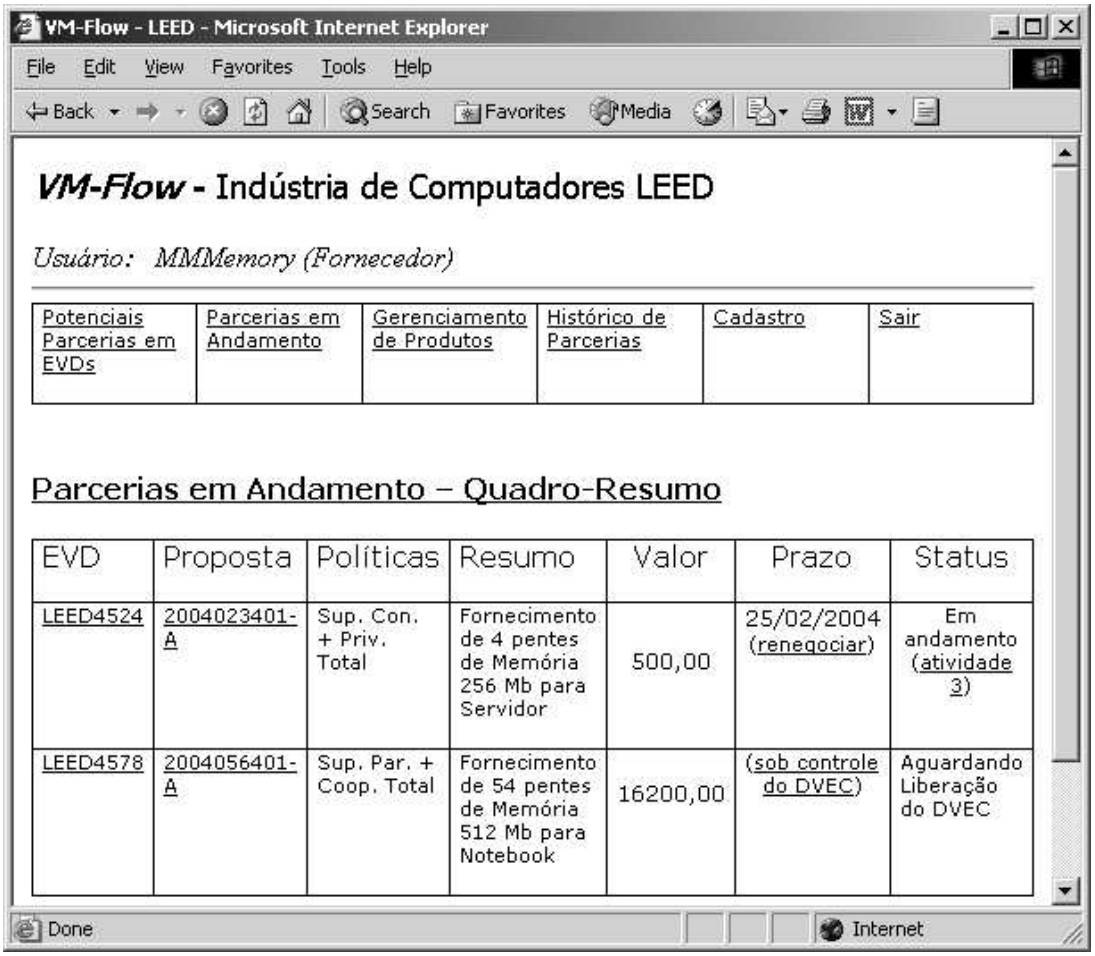
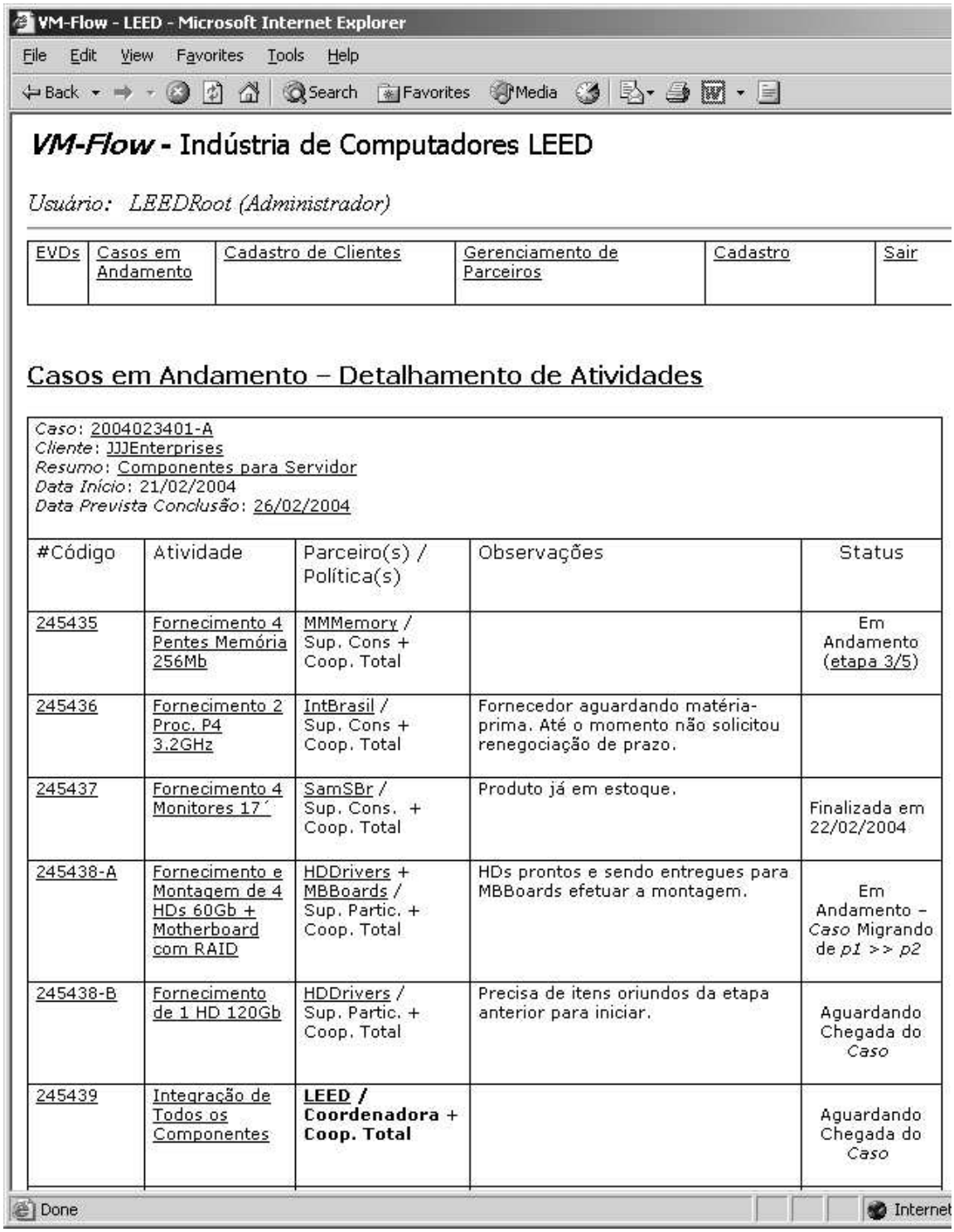


Figura 5.7: LEED - Fornecedor - Parcerias em Andamento



VM-Flow - LEED - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites Media

VM-Flow - Indústria de Computadores LEED

Usuário: LEEDRoot (Administrador)

EVDs	Casos em Andamento	Cadastro de Clientes	Gerenciamento de Parceiros	Cadastro	Sair

Casos em Andamento – Detalhamento de Atividades

Caso: 2004023401-A
Cliente: JJJEnterprises
Resumo: Componentes para Servidor
Data Início: 21/02/2004
Data Prevista Conclusão: 26/02/2004

#Código	Atividade	Parceiro(s) / Política(s)	Observações	Status
245435	Fornecimento 4 Pentes Memória 256Mb	MMMemory / Sup. Cons + Coop. Total		Em Andamento (etapa 3/5)
245436	Fornecimento 2 Proc. P4 3.2GHz	IntBrasil / Sup. Cons + Coop. Total	Fornecedor aguardando matéria-prima. Até o momento não solicitou renegociação de prazo.	
245437	Fornecimento 4 Monitores 17´	SamSBr / Sup. Cons. + Coop. Total	Produto já em estoque.	Finalizada em 22/02/2004
245438-A	Fornecimento e Montagem de 4 HDs 60Gb + Motherboard com RAID	HDDrivers + MBBoards / Sup. Partic. + Coop. Total	HDs prontos e sendo entregues para MBBoards efetuar a montagem.	Em Andamento – *Caso* Migrando de *p1 >> p2*
245438-B	Fornecimento de 1 HD 120Gb	HDDrivers / Sup. Partic. + Coop. Total	Precisa de itens oriundos da etapa anterior para iniciar.	Aguardando Chegada do *Caso*
245439	Integração de Todos os Componentes	**LEED / Coordenadora + Coop. Total**		Aguardando Chegada do *Caso*

Done Internet

Figura 5.8: LEED - Administrador - Visão Global de um Caso

Capítulo 6

Trabalhos Relacionados

Neste capítulo são apresentados alguns trabalhos que são diretamente relacionados com esta dissertação, na área de Empresas Virtuais, Sistemas Interorganizacionais, Sistemas de *Workflow* e também Composição de Serviços. Quando relevante, os pontos comuns e as diferenças com relação ao *VM-Flow* também são analisados.

6.1 Empresas Virtuais

Apesar da existência de diversos outros trabalhos na área [APC02, CMA99, TBV02], o apresentado a seguir é o que mais está relacionado com a plataforma *VM-Flow* e por isso foi objeto de uma análise mais detalhada.

DIVE

O *framework* DIVE (Agent-based Life Cycle Management for Dynamic Virtual Enterprises) [OT01] propõe uma abordagem baseada em ciclo de vida e em agentes para o gerenciamento das EVDs.

De acordo com a ISO [ISO91], um ciclo de vida pode ser definido como "os passos finitos que um sistema pode atravessar durante toda sua existência". O modelo de ciclo de vida proposto pelo *framework* DIVE corresponde ao apresentado na Seção 2.4 desta dissertação:

- **Fase de Especificação e Registro do Processo de Negócio**: durante essa fase os diferentes domínios de negócio devem especificar seus processos de negócio locais e remotos. A especificação dos processos de negócio é realizada através de uma linguagem de definição de processos de negócio. Em seguida, todo domínio registra para cada processo local suas ofertas correspondentes no *marketplace*, expondo as condições nas quais esses processos locais serão oferecidos a outros domínios.

- **Fase de Gerenciamento do Processo de Negócio**: durante essa fase, um domínio de negócio oferece serviços a clientes empregando o modelo dinâmico do *marketplace*. Sempre que um cliente solicita um serviço, o domínio inicia a provisão do mesmo. Se o serviço solicitado for composto por sub-processos remotos que devem ser executados por outros domínios, então o domínio de negócio aciona o *marketplace*, localizando os parceiros potenciais, negociando com eles dinamicamente e finalmente estabelecendo as parcerias para a execução do serviço. Todo esse processo é transparente ao cliente. Durante a execução do serviço, o cliente pode gerenciar o mesmo, ou seja, suspender, reiniciar ou mesmo terminar sua execução. Cada pedido de gerenciamento do cliente deve ser executado de forma autônoma, distribuída e interorganizacional.

Um processo de negócio em DIVE é uma estrutura hierárquica que consiste de um processo raiz e de uma árvore de sub-processos. Os sub-processos que não podem ser divididos são chamados de tarefas - estas são as atividades realmente executadas por um *Resource Business Object* (objeto que representa um recurso - um componente de software, um dispositivo ou ainda um robô).

De forma a executar as principais funcionalidades apresentadas nas duas fases do ciclo de vida, um conjunto de agentes é proposto:

- *RPA (Resource Provider Agent)*: cada *Resource Business Object* é encapsulado e representado por um RPA.
- *PUA (Personal User Agent)*: representante da Empresa Virtual perante os clientes. Cada pedido no DIVE é administrado por um PUA dedicado.
- *WPAs (Workflow Provider Agents)*: conjunto de agentes responsável por instanciar, interpretar e executar os processos de negócio.
- *RNA (Requestor Negotiator Agent)*: quando um sub-processo é especificado como remoto, o WPA responsável cria um RNA e o envia para o *marketplace* para que este localize potenciais parceiros para executar este sub-processo.
- *PNA (Provider Negotiator Agent)*: agentes de negociação localizados em cada um dos candidatos a parceiros das empresas virtuais.
- *STA (Service Type Agent)*: responsável pelo gerenciamento dos tipos de serviços, incluindo seu registro, remoção, listagem e modificação.
- *SOA (Service Offer Agent)*: responsável pelo gerenciamento das ofertas de serviço.

- *SOR (Service Offer Retrieval Agent)*: responsável por selecionar as ofertas associadas a um tipo de serviço de acordo com as restrições dadas pelo requisitante.

Os seguintes domínios administrativos participam da execução e gerenciamento da plataforma:

- *Domínio do Cliente*: domínio do cliente que requisita os serviços da empresa virtual.
- *Domínio do Representante da Empresa Virtual*: este é o domínio através do qual o cliente faz suas requisições e interage com a empresa virtual.
- *Domínio dos Parceiros da Empresa Virtual*: domínio que oferece um conjunto de processos de negócio ao *marketplace* para uma potencial cooperação com outros domínios.
- *Domínio do Marketplace Virtual*: domínio de uma terceira entidade, que oferece os *templates* para os candidatos a parceiros usarem e registrarem suas ofertas.

O *framework* DIVE foi implementado e testado com o auxílio de tecnologias abertas e interoperáveis: os agentes foram desenvolvidos em JAVA sobre Grasshoper [IKV] com as extensões OMG-MASIF e FIPA; para o armazenamento persistente de ofertas de processos de negócio, contratos e especificações de processos, repositórios dedicados baseados em XML foram desenvolvidos.

DIVE x VM-Flow

A Tabela 6.1 apresenta uma análise comparativa entre o *framework* DIVE e a plataforma *VM-Flow*.

Tabela 6.1: DIVE x VM-Flow

——	**DIVE**	**VM-Flow**
Gerenciamento do Marketplace	Agentes Móveis	Workflow/Casos Móveis
Gerenciamento das EVDs	Agentes Móveis	Orquestração/Coreografia
Escalabilidade	Sim	Sim
Políticas de Interação	Não	Sim
Aspectos Interorganizacionais	Parcialmente	Sim
Facilidade de Integração	XML/FIPA/Java	Serviço Web/XML/Java

Apesar de abordagens diferentes, o modelo de ciclo de vida proposto por DIVE foi de muita valia para o entendimento do funcionamento das EVDs e serviu como ponto de partida para a proposição do modelo de gerenciamento da plataforma *VM-Flow*.

6.2 Integração de Sistemas Interorganizacionais

O trabalho apresentado a seguir é, na área de Integração de Sistemas Interorganizacionais, o mais relacionado com esta dissertação.

BPFA

O trabalho apresentado em [SO01] aborda questões arquiteturais referentes às interações B2B (*Business-to-business*) interorganizacionais. [SO01] propõe um modelo de classificação dos processos de negócio em *processos privados* (internos às organizações) e em *processos compartilhados* (que interconectam diferentes organizações). Os processos privados podem expor pontos de interação e os processos compartilhados podem se conectar a estes pontos, de forma que um processo de negócio possa englobar duas ou mais organizações.

Um *framework* que suporte estes dois tipos de processos, o BPFA (*Business Process Framework Architecture*), também é apresentado por [SO01]. O BPFA consiste de um conjunto de componentes que executam instâncias de um modelo de processo interorganizacional, estendendo a infra-estrutura de *workflow* existente em uma companhia e permitindo comunicação orientada a processos entre seus parceiros e clientes.

BPFA x VM-Flow

A Tabela 6.2 apresenta um comparativo entre o framework *BPFA* e o *VM-Flow*.

Tabela 6.2: BPFA x VM-Flow

—	**BPFA**	**VM-Flow**
Suporte à EVs	Parcial	Sim
Suporte à EVDs	Não	Sim
Núcleo	Baseado em Workflow	Baseado em Workflow
Interações Interorg.	Responsabilidade dos SGWFs	Orquestração/Coreografia
Privac. e Autonomia	Proc. Privados e Compartilhados	Políticas de Interação

A categorização proposta por [SO01] dos processos em privados e compartilhados foi o ponto de partida para a proposição das políticas de Interação do *VM-Flow*. Diferentemente do BPFA, a opção de implementar as interações interorganizacionais no *VM-Flow* através de Serviços Web com orquestração e coreografia representou uma maior flexibilidade e também um menor grau de acoplamento entre os sistemas internos de cada parceiro.

6.3 Sistemas de Workflow

O trabalho a seguir foi escolhido na área de Sistemas de *Workflow* por introduzir o gerenciamento descentralizado dos processos de negócio através de casos móveis, conceito aplicado na plataforma *VM-Flow*.

WONDER

Sistemas de Gerenciamento de *Workflow* (SGWFs) tradicionais possuem uma limitação intrínseca de escalabilidade, o servidor central, que representa um gargalo de desempenho e um ponto de falhas em sistemas onde um grande número de casos simultâneos precisa ser executado [SF00]. Com base nesta deficiência dos SGWFs tradicionais, é proposto o projeto e a especificação de uma arquitetura distribuída, utilizando as funcionalidades do ambiente aberto de distribuição CORBA, de forma a suportar, em primeiro lugar, os requisitos de escalabilidade, disponibilidade e confiabilidade dos SGWFs de larga escala. Esta arquitetura, denominada WONDER [SF00], utiliza a idéia de casos (instâncias de processos) móveis que migram pelos nós do sistema, seguindo o plano do processo, conforme as atividades do *workflow* são realizadas. Nos testes realizados por este trabalho, as configurações distribuídas tiveram maior desempenho que as centralizadas para instâncias envolvendo a execução simultânea de mais de 5 casos concorrentes.

Apesar de objetivos bem diferentes, um quadro comparativo entre WONDER e *VM-Flow* é apresentado na Tabela 6.3.

Tabela 6.3: WONDER x VM-Flow

—	**WONDER**	**VM-Flow**
Objetivo Principal	SGWF de Larga Escala	Marketplace Virtual
Núcleo	Baseado em Workflow	Baseado em Workflow
Objetos Distribuídos	CORBA	Java/RMI
Interações Interorg.	Não	Sim
Controle Descentralizado	Sim (Casos Móveis)	Sim (Casos Móveis)

Conforme já mencionado, a plataforma *VM-Flow* implementa o conceito de controle descentralizado baseado em casos móveis proposto pela arquitetura WONDER.

6.4 Composição de Serviços

O trabalho a seguir [Pel03], faz um levantamento sobre orquestração e coreografia de Serviços Web. No contexto desta dissertação, este trabalho foi fundamental para uma

compreensão inicial dos mecanismos de composição de serviços e pela posterior decisão de utilizá-los na plataforma *VM-Flow*.

Orquestração e Coreografia: Tecnologias, Ferramentas e Padrões

De acordo com [Pel03], em uma pesquisa recente realizada entre executivos experientes na área de TI, estes identificaram '*conectar aos clientes, fornecedores e parceiros eletronicamente*' como a questão mais importante atualmente em gerenciamento de TI. Ainda segundo [Pel03], existe uma clara necessidade de estender os negócios atuais de forma a oferecer suporte para a integração de Serviços Web, tanto internos quanto externos. Entretanto, os métodos existentes para criar processos de negócio não foram desenhados para interagir com componentes que cruzassem as barreiras de uma organização. Estes métodos também não são flexíveis o suficiente para lidar com as interfaces introduzidas pelos Serviços Web (SOAP e WSDL, por exemplo). Os processos de negócio atuais necessitam de agilidade para rapidamente se adaptar às necessidades dos clientes e dos mercados, incluindo a incorporação dinâmica de novos clientes, parceiros e fornecedores nos processos de negócio existentes. Neste contexto, segundo [Pel03], os mecanismos de orquestração e coreografia representam uma abordagem aberta e baseada em padrões para conectar Serviços Web e criar processos de negócio de mais alto nível.

As principais tecnologias, ferramentas e padrões relacionados com Composição de Serviços Web (através de Orquestração e Coreografia) são apresentadas e discutidas.

Capítulo 7

Conclusão

As Empresas Virtuais surgiram com a necessidade das companhias em aumentar sua competitividade através da formação de parcerias, reduzindo custos e compartilhando recursos. O avanço nas TICs permitiu o surgimento das Empresas Virtuais Dinâmicas, com ciclos de vida mais curtos e mais adaptadas às mudanças constantes no mercado.

Esta dissertação contribui com a apresentação, implementação e discussão da plataforma *VM-Flow*, uma infra-estrutura de *Marketplace Virtual* que suporta Empresas Virtuais Dinâmicas e é baseada em mecanismos de workflow.

O modelo da plataforma *VM-Flow* é composto de um conjunto de *Facilidades* (importantes no gerenciamento das EVDs) e também de alguns *Serviços de Suporte*. Algumas considerações finais sobre o modelo:

- A facilidade na adequação a diferentes regras e necessidades impostas pelo mercado torna o modelo flexível, e a opção por modelar o núcleo da plataforma utilizando técnicas de Orientação a Objetos aumenta seu grau de extensibilidade.

- A implantação de um mecanismo de gerenciamento descentralizado dos processos de negócio (através dos *casos* móveis) e o projeto do núcleo dando suporte a configurações tanto locais quanto distribuídas tornam o *VM-Flow* uma infraestrutura facilmente escalável.

Diferentes níveis de *privacidade* e *autonomia* são oferecidos aos parceiros (membros das EVDs), através da proposição de um conjunto de *Políticas de Autonomia* e de *Cooperação*, outra contribuição importante desta dissertação.

A combinação destas diferentes políticas dentro de uma mesma EVD tornou-se possível com o auxílio do paradigma de Computação Orientada a Serviços, mais especificamente do uso de Serviços Web e de Orquestração e Coreografia.

A aplicação da especificação BPEL4WS (e do motor de orquestração BPWS4J) como mecanismo para compor os serviços foi muito bem sucedida, demonstrando o potencial dessa especificação.

Uma aplicação-exemplo construída sobre o *VM-Flow* (indústria de computadores LEED), também é apresentada como forma de demonstrar a utilização prática da infra-estrutura. Nesta aplicação, a indústria hipotética de computadores LEED usa o *VM-Flow* para encontrar parceiros, formando EVDs com o objetivo da fabricar os produtos (Desktops, Servidores e Notebooks) encomendados por seus clientes (outras empresas em geral).

Trabalhos Futuros

Possíveis extensões a este trabalho incluem:

- *Infraestrutura*:
  - Especialização das facilidades.
  - Implantação de mecanismos de negociação de contratos.
  - Agregação de novos serviços de suporte.
  - Adaptação do MPCI para aplicações móveis.
- *Políticas*:
  - Adequação das Políticas existentes para nichos de mercado específicos.
  - Proposição de Novas Políticas.

Referências Bibliográficas

[AHK+96] L. M. Applegate, C. W. Holsapple, R. Kalakota, F. J. Radermacher, e A. B. Whinston. Electronic commerce: Building blocks of new business opportunity. *Journal of Organizational Computing and Electronic Commerce*, 6(1):1–10, 1996.

[APC02] P. Avila, G.D. Putnik, e M. M. Cunha. Brokerage function in agile/virtual enterprise integration - a literature review. Em *3rd IFIP Working Conference on Infrastructures for Virtual Enterprises (PRO-VEt02)*, páginas 65–72, Portugal, Maio 2002. Kluwer Academic Publishers.

[BEK+00] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H.F. Nielsen, S. Thatte, e D. Winer. *SOAP: Simple Object Access Protocol*. DevelopMentor, IBM Corporation, Lotus Development Corporation, Microsoft and UserLand Software, http://static.userland.com/xmlRpcCom/soap/SOAPv11.htm, Abril 2000.

[BLFM98] T. Berners-Lee, R. Fielding, e L. Masinter. *Uniform Resource Identifiers (URI): Generic Syntax - RFC 2396*. Internet Engineering Task Force (IETF), http://www.ietf.org/rfc/rfc2396.txt, 1998.

[CKM+03] F. Curbera, R. Khalaf, N. Mukhi, S. Tai, e S. Weerawarana. Service-oriented computing. *Communications of the ACM*, 46(10):25–28, Outubro 2003.

[CMA99] L. M. Camarinha-Matos e H. Afsarmanesh. The virtual enterprise concept. Em *IFIP International Conference on Infrastructures for Virtual Enterprises (PRO-VE 99)*, páginas 3–14, Porto, Portugal, Outubro 1999. Kluwer Academic Publishers.

[DH98] Y.L. Doz e G. Hamel. *Alliance Advantage. The Art of Creating Value through Partnering*. Boston: Harvard Business School Press, 1998.

[F96] Lin F. *Reengineering the Order Fulfillment Process in Supply Chain Networks: A Mutliagent Information Systems Approach.* Tese de Doutorado, University of Illinois at Urbana-Champaign, 1996.

[Fie98] R. Fielding. Support for the virtual enterprise: web-based development of complex information products. *Communications of the ACM*, 41(8):84–92, Agosto 1998.

[HF86] A. B. Holanda Ferreira. *Novo Dicionário da Língua Portuguesa.* Editora Nova Fronteira, 2a. edição edição, 1986.

[Hol94] D. Hollingsworth. Reference model - revision 1.1. Relatório Técnico WfMC-TC00-1003, Workflow Management Coalition, http://www.aiim.org/WfMC/standards/docs/tc003v11.pdf, 1994.

[IBM03] IBM. *Business Process Execution Language for Web Services Java Run Time.* http://alphaworks.ibm.com/tech/bpws4j, 2003.

[IKV] IKV++. *Grasshopper.* http://www.grasshoper.de.

[ISO91] ISO/IEC. *Common Management Information Protocol (CMIP).* 9596, Information Technology, Open Systems Interconnection, Genebra, Suíça, 1991.

[Mer01] D. Mertz. Comparing w3c xml schemas and document type definitions (dtds). Relatório Técnico, Gnosis Software, Inc, http://www-106.ibm.com/developerworks/xml/library/x-matters7.html, 2001.

[OAS00] OASIS. *UDDI Technical White Paper.* UDDI.org, http://www.uddi.org/pubs/Iru_UDDI_Technical_White_Paper.pdf, Setembro 2000.

[OMG02] Object Management Group OMG. *CORBA 3.0 - The Common Object Request Broker: Architecture and Specification.* http://www.omg.org/cgi-bin/apps/do_doc?formal/02-06-01.pdf, Julho 2002.

[OT98] V. Ouzounis e V. Tschammer. Integration of electronic commerce business processes in virtual enterprises. Em *European Multimedia, Microprocessor Systems and Electronic Commerce Conference and Exposition (EMMSEC 98), Bordeaux, França*, Setembro 1998.

[OT99] V. Ouzounis e V. Tschammer. A framework for virtual enterprise support services. Em *32nd Hawaii Int'l Conference on System Sciences (HICSS-32), Maui, Hawaii*, volume 8, página 8023, Janeiro 1999.

[OT01] V. Ouzounis e V. Tschammer. Towards dynamic virtual enterprises. Em *IFIP Conference on Towards The E-Society: E-Commerce, E-Business, E-Government (I3E 2001)*, páginas 177–192, Zurique, Suiça, 2001. Kluwer Academic Publishers.

[Ouz98] V. Ouzounis. *Electronic Commerce and New Ways of Work - An R&D Road-Map*. European Commission - Directoral General III, 1998.

[Ouz01] V. Ouzounis. *An Agent-Based Platform for the Management of Dynamic Virtual Enterprises*. Tese de Doutorado, Technische Universität Berlin, 2001.

[Pel03] C. Peltz. Web services orchestration - a review of emerging technologies, tools, and standards. Relatório Técnico, Hewlett Packard, 2003.

[PG03] M.P. Papazoglou e D. Georgakopoulos. Service-oriented computing. *Communications of the ACM*, 46(10):25–28, Outubro 2003.

[Pro98] Crossflow Project. *ESPRIT Project 28635 Cross-Organisational Workflow Management*. The Crossflow Consortium, http://www.crossflow.org, 1998.

[SF00] R.S. Silva Filho. Uma arquitetura baseada em corba para workflow de larga escala. Dissertação de Mestrado, UNICAMP - Universidade Estadual de Campinas, 2000.

[SIM02] BEA Systems, IBM, e Microsoft. *Web Services Transaction (WS-Transaction)*. ftp://www6.software.ibm.com/software/developer/library/ws-transpec.pdf, Agosto 2002.

[SIM03a] BEA Systems, IBM, e Microsoft. *Web Services Coordination (WS-Coordination)*. ftp://www6.software.ibm.com/software/developer/library/ws-coordination.pdf, Setembro 2003.

[SIM+03b] BEA Systems, IBM, Microsoft, SAP AG, e Siebel Systems. *Business Process Execution Language for Web Services (BPEL4WS) - Version 1.1*. http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/, 2003.

[SISM02] BEA Systems, Intalio, SAP, e Sun Microsystems. *Web Service Choreography Interface 1.0*. http://www.sun.com/software/xml/developers/wsci/wsci-spec-10.pdf, 2002.

[SM03] I.J.G. Santos e E.R.M. Madeira. Policy-based orchestration and choreography of services on dynamic virtual enterprises. Em *3rd IFIP Conference on E-business, E-commerce and E-government*, Research Colloquium, Setembro 2003.

[SO01] K. Schulz e M. Orlowska. Architectural issues for cross-organisational b2b interactions. Em *International Workshop on Distributed Dynamic Multiservice Architectures (DDMA)*, EUA, Abril 2001. IEEE Computer Society Press.

[SRKT00] C. Stricker, S. Riboni, M. Kradolfer, e J. Taylor. Market-based workflow management for supply chains of services. Em *33rd Hawaii Int'l Conference on System Sciences (HICSS-33), Maui, Hawaii*, volume 6, página 6022, Janeiro 2000.

[TBV02] M. Tolle, P. Bernus, e J. Vesterager. Reference models for virtual enterprises. Em *3rd IFIP Working Conference on Infrastructures for Virtual Enterprises (PRO-VE'02)*, páginas 3–10, Portugal, Maio 2002. Kluwer Academic Publishers.

[Vel03] R.R. Veloso. *Java e XML - Processamento de documentos XML com Java.* Novatec Editora, 2003.

[W3C99] World Wide Web Consortium W3C. *Namespaces in XML.* W3C Recomendation, http://www.w3.org/TR/1999/REC-xml-names-19990114, Janeiro 1999.

[W3C00] World Wide Web Consortium W3C. *Extensible Markup Language (XML) 1.0.* W3C Recommendation, http://www.w3.org/TR/REC-xml, segunda edição, 2000.

[W3C03a] World Wide Web Consortium W3C. *Web Services Description Language (WSDL) Version 1.2.* W3C Working Draft, http://www.w3.org/TR/wsdl12/, 2003.

[W3C03b] World Wide Web Consortium W3C. *SOAP Version 1.2 Part 0: Primer.* W3C Recomendation, http://www.w3.org/TR/2003/REC-soap12-part0-20030624/, Junho 2003.

[ZP99] A. Zarli e P. Poyet. A framework for distributed information management in the virtual enterprise. Em *IFIP TC5 WG5.3 / PRODNET Working Conference for Virtual Enterprises (PRO-VE'99)*, páginas 293–306, 1999.