Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e de Computação
Departamento de Computação e Automação Industrial

Uma Abordagem Unificada para Modelar Processos de Work-flow e seu Software de Suporte

José Adalberto Soto Mejía

Tese de Doutorado apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas, como parte dos requisitos exigidos para a obtenção do título de Doutor em Engenharia Elétrica e de Computação

Banca Examinadora:

Prof. Dr. Manuel de Jesus Mendes (orientador)
Prof. Dr. Marco Aurelio Amaral Henriques
Prof. Dr. Rafael Santos Mendes
Prof. Dr. Juan Manuel Adán Coello
Prof. Dr. Luiz Ary Messina

Campinas, São Paulo, Junho 06 de 2002

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

So78u

Soto Mejía, José Adalberto
Uma abordagem unificada para modelar processos de workflow e seu software de suporte / José Adalberto Soto Mejía. --Campinas, SP: [s.n.], 2002.

Orientador: Manuel de Jesus Mendes.
Tese (doutorado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Modelos e construção de modelos. 2. Engenharia de software auxiliada por computador. 3. Engenharia de Software. I. Mendes, Manuel de Jesus. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Resumo

A modelagem dos processos de workflow e das especificações do seu software de suporte têm sido abordadas por analistas de processos de negócio e por desenvolvedores do software de suporte de maneira independente, baseados em metamodelos diferentes. Essa situação tem criado problemas de comunicação entre eles, tornando difícil que as descrições dos processos de workflow e as especificações técnicas do seu software de suporte sejam entendidas por ambas as partes. Como conseqüência, com freqüência a operação dos sistemas de apoio à automação dos processos de negócio não está diretamente relacionada com os processos de workflow que ela suporta.

Esta situação justifica a importância de uma abordagem unificada para modelar os processos de workflow e seu software de suporte. Sob um ponto de vista conceitual, a proposta de um metamodelo unificado, para modelar ambos aspectos, é um dos objetivos centrais desta tese. Um outro objetivo propõe a representação plausível de uma definição de processo de workflow, através de uma notação textual que seja consistente com o metamodelo unificado proposto. A notação textual proposta visa transformar uma representação gráfica de uma definição de processo de workflow numa representação computacionalmente interpretável por uma máquina de workflow.

Sob um ponto de vista de implementação, e baseado nos resultados conceituais mencionados, foi implementado em Java e CORBA um protótipo de máquina de workflow, que interpreta definições de processoS de workflow expressas na notação textual proposta. A máquina de workflow foi integrada num ambiente de execução que fornece funcionalidades básicas para o desenvolvimento de um mercado de serviços, a plataforma Platin (Platin Middleware Platform). A máquina de workflow foi usada para combinar e coordenar serviços integrados na plataforma mencionada e os resultados da execução são ilustrados.

Palavras chaves:

Definição de processo de workflow, Metamodelo, Sistema de Computação Empresarial, EDOC, perfil de UML.

Abstract

Business analysts and software developers have approached the modeling of workflow processes and the specification of the software that supports them in an independent way using different metamodels. This situation is a source of misunderstanding and it makes difficult for both parts the correct understanding of the workflow process definitions and the specifications of the software that supports them. As a consequence, the system operation is frequently not related with the workflow processes that it pretends to support.

The above-described situation justifies the importance of an unified approach for modeling both the workflow processes and the software systems that support them. A unified metamodel for modeling both mentioned aspects is, from a conceitual view point, one of the main objectives of this dissertation. Another objetive is to propose a plausible representation of a workflow process definition in terms of a textual notation that is consistent with the proposed unified metamodel. The proposed textual notation is a plausible approach to transform a graphical workflow process definition into a representation computationally interpretable by a workflow engine.

From an implementation view point, and based in the above mentioned theoretical structure, a simple prototype of a workflow engine that interprets and executes a workflow process definition coded in the textual notation proposed was implemented using CORBA and the Java language. The workflow engine was deployed into an execution environment that offers basic functionality for an open service market place (The Platin Middleware Platform). The workflow engine was used to combine and coordinate services deployed in the mentioned platform and the execution results are illustrated.

Key words:

Workflow process definition, Metamodel, Enterprise Computing System, EDOC, UML profile

Agradecimentos

A todas as pessoas que me ajudaram durante o desenvolvimento deste trabalho; aos amigos do Cenpra, Paulo, Domingos, Jorge, Rubens, por sua companhia e apoio; a todas as instituições que de uma forma o outra têm me prestado suporte para a realização deste trabalho.

Em particular, um especial agradecimento:

- ao meu orientador Dr. Manuel de Jesus Mendes por seu exemplo de trabalho acadêmico, orientação durante o desenvolvimento da tese, confiança e apoio oferecido nos momentos difíceis;
- à Universidade Estadual de Campinas, UNICAMP;
- ao Conselho Nacional de Desenvolvimento Científico e Tecnológico, CNPq;
- à Universidad Tecnológica de Pereira, UTP;
- ao Centro de Pesquisas Renato Archer, CenPRA;
- ao Research Institute for Open Communication Systems, FOKUS.

Dedicatória

A meus pais, José e Ligia

Índice Geral

Índice de Figuras

Índice de Tabelas

CAPÍTULO 1

INTRODUÇÃO GERAL

Os sistemas de workflow são essenciais nas organizações que necessitam automatizar seus processos de negócio. Eles permitem às organizações especificar, executar e monitorar seus processos de negócio através das redes de computação empresarial, o que tem como resultado uma melhoria nos resultados do processo, uma melhor utilização dos recursos organizacionais e um monitoramento mais eficiente do processo.

A modelagem dos processos de negócio de uma maneira formal e precisa é necessária, não só para automatizá-los, usando a tecnologia da informação, mas também para fazer a sua reengenharia. Atualmente estão disponíveis mais de 40 tecnologias diferentes de diagramas de modelagem para expressar as estruturas dos negócios e seus processos. A maioria delas foi desenvolvida por especialistas num domínio específico, para responder a particularidades da área, usando variações das tecnologia já existentes e adaptando-as às respectivas necessidades. KNUTILLA et al. apresentam em [1.1] uma análise detalhada, avaliando várias representações de processos (metodologias, ferramentas, padrões e linguagens). No documento citado, 31 linguagens[1] de especificação de processos foram avaliadas tomando como referência um conjunto de requerimentos identificados previamente como necessários no domínio dos processos de manufatura (planejamento da produção, planejamento dos processos de manufatura e workflow). As representações analisadas, embora não sejam uma lista exaustiva de todas as representações de processos atualmente disponíveis, são uma boa amostra que providencia uma perspectiva sobre as diferentes formas de representar a informação de processos.

A análise dessas representações permite concluir que:

- existem várias e diferentes maneiras de satisfazer os requerimentos necessários para representar processos;
- algumas das representações (Redes de Petri, por exemplo) são enfoques gerais, enquanto que outras são mais especializadas e relativas a domínios específicos de aplicação;
- por sua própria natureza, o enfoque genérico é aplicável mais amplamente e é mais flexível, mas requer a introdução de construtores adicionais para captar alguns dos requerimentos mais específicos;
- quase todas as representações dão ênfase à sintaxe da especificação e não à sua semântica ou significado dos termos;

[1] ACT, ALPS, AP213, Behavior Diagrams, Core Plan Representation, Entity Relationship, Functional Flow Block Diagrams, Gantt Charts, Generalized Activity Network, Hierarchical Task Networks, IDEF0, IDEF3, <I-N-O-V-A>, Knowledge Interchange Format, O-Plan, OZONE, PAR2, Part 49, PERT Networks, Petri Nets, Process Flow Representation, Process Interchange Format, Quirk Model, Visual Process Modeling Language, AND/OR Graphs, Data Flow Diagrams, Directed Graphs, State Transition Diagrams, Tree Structures

- intercâmbio de modelos de processo entre diferentes domínios de aplicação cria situações nas quais os mesmos termos podem ter significados diferentes.

Para a especificação particular de processos de workflow, existem também várias linguagens. Em geral cada fornecedor de um sistema de workflow baseia-se num formato proprietário para a especificação do modelo do processo de workflow, e só alguns fazem uso (no componente para modelagem de seus produtos) das linguagens já existentes de modelagem de processos, como por exemplo, Petri-Nets [1.9].

Os múltiplos enfoques acima mencionados conduzem a sérios problemas de interoperabilidade, não só para os usuários que utilizam vários produtos e que seguem diferentes paradigmas de modelagem, como também para aqueles que tentam integrar seus processos de negócio com passos prévios da cadeia produtiva (como fornecedores e clientes, por exemplo).

Com o intuito de estimular o uso da tecnologia de workflow, procurando resolver os problemas de interoperabilidade acima mencionados, criou-se , nos primeiros anos da tecnologia de workflow (1994), a "Coalizão para Gerenciamento de Workflow" (Workflow Management Coalition, WfMC) [1.2]. A WfMC é uma organização não lucrativa de fornecedores de produtos de workflow, usuários, consultores e pesquisadores, que lidera os esforços para resolver os problemas de interoperabilidade entre os diferentes produtos de workflow, através de padrões próprios. Os primeiros esforços da WfMC consideraram:

- a padronização dos termos usados no contexto de workflow [1.3],
- a abstração, a partir das aplicações com workflow e dos sistemas de gerenciamento de workflow existentes na época, de uma arquitetura comum a eles, conhecida como "O Modelo de Referência de Workflow da WfMC" (The WfMC's Reference Model) [1.4], arquitetura esta baseada no modelo de dois níveis cliente-servidor e
- a formalização das interfaces entre os componentes da arquitetura [1.4], [1.5], [1.6], [1.7], [1.8].

Por representar o "Modelo de Referência de Workflow da WfMC" uma excelente visão panorâmica da arquitetura das aplicações baseadas em workflow e dos respectivos sistemas de gerenciamento de workflow e, por ser a terminologia desta arquitetura [1.3] de ampla aceitação, será apresentada , na seção 1.1 , uma breve síntese de ambos. As definições e o conteúdo dessa seção servirão como referência de contexto para o resto da tese.

Embora a terminologia padronizada pela WfMC continue sendo a de maior aceitação, o seu Modelo de Referência está hoje ultrapassado, na perspectiva de interconectividade de redes. O Modelo de Referência reflete o fato de que os sistemas tradicionais de workflow foram projetados para um ambiente homogêneo, no qual todos os workflows seriam especificados num único modelo proprietário de especificação de workflow, executado-se também num ambiente proprietário e utilizando recursos humanos e computacionais registrados num único serviço de diretório.

O atual ambiente de redes globais e conectividade entre várias empresas tem conduzido à necessidade de especificar novos padrões que enfoquem os *problemas de execução* de workflows descentralizados e distribuídos via componentes, através da infra estrutura da rede global. Os esforços nesse sentido têm sido liderados pelo "Grupo para o Gerenciamento de Objetos"- OMG[2] (Object Management Group, OMG) [1.10] com a padronização, em Julho de 1999, de um sistema de interfaces de execução para a Funcionalidade de Workflow conhecido como Workflow Management Facility [1.11] e que faz parte da arquitetura OMA[3] - (Object Management Architecture- OMA) [1.12] da OMG.

O trabalho na OMG para padronizar a modelagem de sistemas orientados a objetos, tem-se concentrado em definir a Linguagem de Modelagem Unificada (Unified Modeling Language- UML) [1.13] como padrão. Consistente com a UML, outro esforço de padronização dentro da OMG está orientado a prover padrões que suportem o enfoque orientado a objetos, no projeto de sistemas de computação empresarial, usando a tecnologia de objetos distribuídos. Estes sistemas são conhecidos como sistemas EDOC[4] (Enterprise Distributed Object Computing Systems –EDOC Systems).

Embora a UML, como linguagem genérica, providencie suporte à análise e projeto de sistemas orientados a objetos, a atual especificação de UML (versão 1.4) não enfoca todos os aspectos importantes para orientar a modelagem dos sistemas de computação empresarial num ambiente de computação de objetos empresariais distribuídos- sistemas EDOC.

Embora esses assuntos tenham sido abordados com sucesso (e de várias maneiras) usando a notação básica de UML, era importante padronizar o paradigma do uso de UML no contexto dos sistemas de computação empresarial. Para resolver esta situação a OMG emitiu, em Outubro de 1999, uma Chamada de Propostas solicitando um Perfil de UML para Computação de Objetos Empresariais Distribuídos (UML Profile for Enterprise Distributed Object Computing) [1.14]. Essa Chamada de Propostas está relacionada especificamente com os *aspectos de projeto* de tais sistemas de software, no contexto *do uso da tecnologia de implementação de objetos distribuídos.* Embora o perfil acima solicitado tenha a ver exclusivamente com a *engenharia do software* dos sistemas de computação empresarial, a

[2] OMG-Object Management Group: maior consórcio de software do mundo que procura criar uma arquitetura orientada a objetos baseada em especificações que suportem objetos distribuídos visando: reuso, portabilidade, e interoperabilidade.

[3] OMA-Object Management Architecture. Inclui:

- ORB (Object Request Broker): é o componente que permite a comunicação entre clientes e objetos num ambiente distribuído. Um ORB é o mecanismo através do qual os objetos, de maneira transparente, fazem requisições a (ou recebem resposta de) outros objetos na mesma máquina, ou através da rede.
- Serviços de Objetos (Object Services): são interfaces a serviços de propósito geral independentes do domínio de aplicação e fundamentais no desenvolvimento de aplicações baseadas em OMA.
- Funcionalidades Comuns (Common Facility): são interfaces para facilidades (funcionalidades) orientadas ao usuário final, aplicáveis na maioria dos domínios. A especificação inclui as interfaces que os diferentes objetos devem suportar para prover e/ou participar da facilidade.

[4] Um Sistema de Computação de Objetos Empresariais Distribuídos (sistema EDOC) é um sistema de software construído usando tecnologia de objetos distribuídos e que providencie, dentro de uma empresa , suporte para a realização de um conjunto integrado de seus processos de negócio.

implementação com sucesso de tais sistemas requer que a operação do sistema esteja diretamente relacionada com os processos de negócio que ele suporta.

A integração entre organizações e seus sistemas de computação empresarial é hoje uma necessidade. Foi mencionado em [1.15] que, numa indústria altamente competitiva , dois dos principais fatores de sucesso são: (i) aquisição e retenção de clientes e (ii) provisão de características de valor agregado e serviços combinados. Esses dois fatores conduzem à necessidade da integração entre organizações, já que os clientes preferem tratar com um só provedor de serviços e a maioria dos serviços com valor agregado requerem a integração do que os diferentes provedores têm para oferecer. Esses processos de negócios virtuais [1.16], [1.17] que operam através de várias empresas, podem ser implementados usando um conjunto de definições de workflow [1.18] criadas para suportar segmentos particulares de todo o processo. Contudo, as definições específicas de workflow devem ser desenvolvidas tendo em vista o seu intercâmbio ou uso em diferentes domínios.

Em geral a especificação de processo, que é desenvolvida para ser intercambiada ou usada em diferentes domínios, deve ter um conjunto de termos semanticamente definidos de maneira não ambígua. O anterior também é válido no caso particular de processos de workflow como foi anotado por BUSSLER em [1.18], ao assinalar que a publicação de uma linguagem de definição de workflow, sem explicitamente definir o metamodelo de workflow subjacente, ocasiona uma indefinição semântica na linguagem, embora ela esteja sintaticamente bem definida.

Enquanto o processo para estender o UML com um perfil para os sistemas EDOC estava em andamento, a OMG, em Dezembro de 2000, publicou uma Chamada de Propostas solicitando extensões ao UML para expressar Definições de Processo de Workflow (UML Extensions for Workflow Process Definition, Request for Proposal) [1.19]. Até a publicação pela OMG dessa Requisição de Propostas, nenhuma das duas organizações (WfMC, OMG) tinha considerado a padronização de um metamodelo de workflow que permitisse aos diferentes sistemas de gerenciamento de workflow intercambiar definições de processos de workflow. O processo solicitando um perfil de UML para Computação de Objetos Empresariais Distribuídos está parcialmente concluído com a aprovação, em Dezembro de 2001, desse perfil na qualidade de ‘draft’ [1.20]. O processo solicitando um perfil de UML para Definições de Processo de Workflow está ainda em andamento (Junho de 2002).

Embora os esforços acima descritos visem estender a UML para padronizar a modelagem de sistemas EDOC, a partir da perspectiva do projeto de um sistema de software, e para resolver problemas básicos e fundamentais de interoperabilidade na área de workflow, no momento de aplicar a tecnologia de workflow a algum dos processos de negócio da empresa, surge um ‘novo tipo de problema de interoperabilidade’ que poderíamos chamar de ‘desentendimento’ entre o modelo da realidade e o modelo da sua automação. O seguinte cenário descreve a natureza desse desentendimento, antes da criação do perfil de UML para os sistemas EDOC.

O consultor de negócios trabalha em conjunto com o usuário para descrever os processos de negócio da corporação que vão ser suportados pela tecnologia de workflow. O grupo de desenvolvimento do 'software' recebe a descrição do consultor, porém tem problemas para

entender a terminologia do negócio e acha a descrição informal demais para ser usada na implementação do sistema (antes da publicação do perfil EDOC, já que agora EDOC suporta a especificação do sistema, desde a análise até à implementação num ambiente de computação distribuído, usando um modelo de componentes de classes empresariais). Os desenvolvedores do software escrevem sua própria especificação sob um ponto de vista técnico. Quando a especificação do sistema é apresentada aos usuários, ela não é entendida completamente, por ser técnica demais e por usar uma terminologia diferente, porém eles são forçados a aceitá-la para poder continuar com o projeto.

A situação anterior facilmente conduz a um sistema que não cumpre com os requisitos do usuário. Os problemas de comunicação entre consultores e desenvolvedores tornam difícil conseguir que as descrições dos processos de negócio e as especificações técnicas do software sejam entendidas por ambas as partes . O certo está em modelar os processos de negócio e os sistemas de software, que dão suporte a eles, de uma maneira que seja precisa (com semântica bem definida) e ao mesmo tempo amistosa (fácil de entender).

A situação problemática acima descrita sugere a importância de propor um metamodelo unificado dos processos de negócio (ou em termos mais explícitos dos processos de workflow associados com os processos de negócio) e dos sistemas de software que dão suporte a eles (aplicação de workflow) com o objetivo de dar alguns passos para resolver o problema que antes chamamos 'problema de desentendimento'. Da natureza desse 'desentendimento' pode-se, em princípio, inferir que o metamodelo para a definição de processos de workflow deve ser consistente semanticamente com o padrão já adotado para especificar sistemas EDOC. Observe-se, contudo, que esse metamodelo não necessariamente virá a considerar a complexidade envolvida com a especificação de um sistema, pela perspectiva do seu projeto num ambiente de computação distribuído e usando um modelo de componentes. A proposta, sob o ponto de vista conceitual, desse metamodelo unificado é o objetivo central desta tese.

Durante a execução de uma aplicação de workflow, a definição do processo de workflow, especificada usando o metamodelo unificado, deve ser interpretado e executado por um software particular conhecido como máquina de workflow (ver seção 1.1). Como um segundo objetivo conceitual, a tese propõe uma notação textual consistente com o metamodelo unificado proposto, que permite que a representação gráfica da definição do processo de workflow possa ser mapeada numa representação computacionalmente interpretável por uma máquina de workflow. Esse enfoque permite que o modelo computacionalmente interpretável, mas ainda genérico no sentido de ser independente da plataforma de execução, possa ser interpretado e executado por uma máquina de workflow implementada para um ambiente específico, usando um componente interpretador que sirva de ponte entre o modelo independente da plataforma e o ambiente de execução específico.

Com base nos resultados desta proposta, um terceiro objetivo do trabalho pretende mostrar, por meio da implementação de uma máquina de workflow e seu correspondente componente interpretador, que é possível interpretar e executar uma definição de processo de workflow em termos da notação textual proposta.

Com o objetivo de ter uma referência básica sobre workflow, que sirva de contexto para o resto da tese, a seção 1.1 apresenta um resumo do "Modelo de Referência de Workflow" e da terminologia básica sobre workflow da WfMC. A seção 1.2 resume os esforços de padronização na OMG relacionados com workflow e esclarece os conceitos de metamodelo (seção 1.2.1), perfil (seção 1.2.2) e a arquitetura de metamodelos da OMG (seção 1.2.3) diretamente relacionados com o conteúdo desta tese. Finalmente a seção 1.3 apresenta a estrutura da tese.

1.1 O MODELO DE REFERÊNCIA DE WORKFLOW DA WfMC E GLOSSÁRIO BÁSICO

Para definir padrões na área de workflow o primeiro passo dado pela WfMC foi a identificação das partes dos Sistemas de Gerenciamento de Workflow (ver definição na Tabela 1.1) e das aplicações baseadas em workflow que deveriam ser padronizadas, para permitir que blocos desenvolvidos independentemente pudessem interagir. Como resultado desse processo foi publicado o 'Modelo de Referência de Workflow da WfMC' ilustrado na Figura 1.1. Esse Modelo identifica a arquitetura comum às aplicações baseadas em workflow e aos sistemas de gerenciamento de workflow, ou seja, os componentes principais dessas aplicações e sistemas e as interfaces abstratas para a interação entre eles. A Figura 1.1, ilustra os principais componentes e interfaces que fazem parte da arquitetura de um workflow.

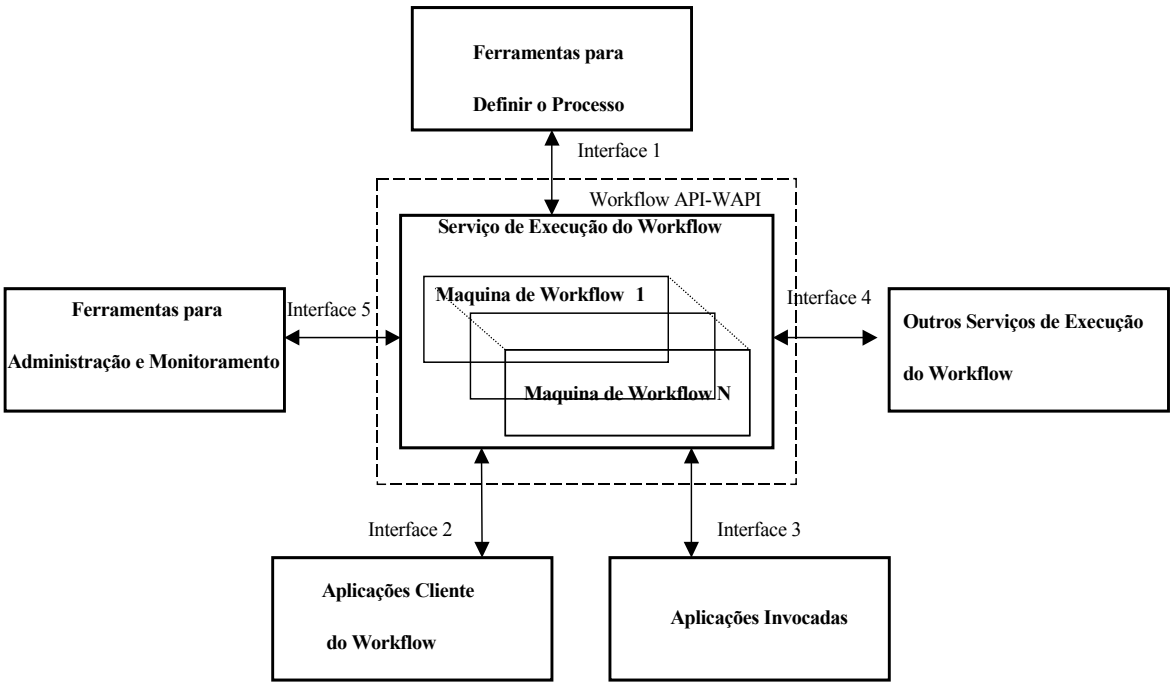


Figura 1.1 O Modelo de Referência de Workflow da WfMC, Componentes e Interfaces.

Para facilitar a Referência a outros documentos, a descrição de cada entidade a ser citada estará acompanhada pela respectiva transcrição ao Inglês, tal como aparece na fonte origi-

nal. Aparecerão sublinhados todos os termos que fazem parte do glossário padronizado pela WfMC e cujas definições foram consignadas na Tabela 1.1 'Terminologia básica sobre workflow'.

O Modelo de Referência da WfMC mostrado na Figura 1.1 define cinco componentes :

- Ferramentas para Definir o Processo (Process Definition Tools): ferramentas usadas para capturar a lógica do processo de negócio numa notação de alto nível de abstração.
- Serviço de Execução do Workflow (Workflow Enactment Service): é o software servidor do workflow e nervo central do sistema de workflow. É responsável pela administração do processo, distribuição e invocação das atividades. Para a execução de instâncias particulares de workflow é usada a máquina de workflow.
- Aplicações Cliente do Workflow (Workflow Client Applications): são aplicações com interface gráfica para o usuário ver e administrar no servidor de workflow (Serviço de Execução do Workflow) o conteúdo de sua lista de trabalho e interagir com seus ítens de trabalho gerenciados pelo servidor de workflow.
- Aplicações Invocadas (Invoked Applications): são as aplicações invocadas pelo servidor de workflow (Serviço de Execução do Workflow) para realizar atividades automatizadas.
- Ferramentas para Administração e Monitoramento (Administration & Monitoring Tools): ferramentas usadas para administrar a execução e monitorar o estado do fluxo de trabalho, através do sistema de workflow.

O Modelo define também interfaces entre esses componentes. O conjunto de interfaces é conhecido como WAPI (WAPI = Workflow Application Programming Interface). A seguir, é dada uma descrição de cada uma delas:

Interface 1: A interface entre a Ferramenta para Definir o Processo e o software de gerenciamento do workflow (Serviço de Execução do Workflow). Define um formato e um conjunto de chamadas ao WAPI para intercambiar as especificações do processo entre as Ferramentas para Definir o Processo e o servidor do workflow. (Interface entre a ferramenta e o servidor de workflow).

Interface 2: Provê o conjunto de interações entre uma Aplicação Cliente do Workflow e o servidor do workflow (Serviço de Execução do Workflow). Estas incluem interação com a lista de trabalho, controle do processo de workflow e funções administrativas. (Interface entre o cliente e o servidor de workflow).

Interface 3: Esta interface descreve como são invocadas as aplicações. (Interface entre o servidor de workflow e as atividades).

Interface 4: Esta interface descreve as interações entre dois servidores de workflow (entre Serviços de Execução do Workflow). As interações incluem iniciação, consulta e controle do processo de workflow e suas atividades e funções administrativas. (Interface entre servidores de workflow).

Interface 5: Provê o conjunto de funções para administrar e monitorar um servidor de workflow. (Interface entre o monitor e o servidor de workflow).

A Tabela 1.1 "Terminologia básica da WfMC relacionada com Workflow", contem as definições dos conceitos básicos na área de workflow, segundo a WfMC. Para facilitar a Referência a outros documentos, a descrição de cada entidade citada está acompanhada pela respectiva transcrição ao Inglês, tal como aparece na fonte original.

Tabela 1.1Terminologia básica da WfMC relacionada com Workflow

Processo de Negócio (Business Process): um processo de negócio é algo que se desenvolve na vida real, definido por passos que conduzem à realização de metas de negócio e por regras que determinam a seqüência dos passos, normalmente dentro do contexto de uma estrutura organizacional que define papeis e relacionamentos. Um processo de negócio pode consistir de atividades automatizáveis e de atividades manuais, as quais ficam fora do âmbito do gerenciamento do workflow. Os processos de negócio existem independentemente dos sistemas de workflow.
Processo (Process): é a *visão formalizada* de um processo de negócio representado como um conjunto coordenado (paralelo/ou serial) de uma ou mais atividades ou procedimentos conectados os quais, coletivamente, realizam um objetivo ou uma meta. As atividades podem ser manuais ou automatizáveis. Pode ser representado por uma rede de atividades ou por um grafo de atividades orientadas ou por uma folha de instruções.
Workflow: a automação de um Processo de Negócio (em sua totalidade ou em parte) durante a qual documentos, informação, tarefas são passadas de um participante ao outro (para que seja realizada uma ação), de conformidade com um conjunto de regras de procedimento. A automação do processo *é definida dentro* de uma Definição de Processo que identifica as atividades, regras de procedimento e dados de controle, usados para administrar o workflow durante a execução do processo.
Definição de Processo (Process Definition): a *representação* de um processo de negócio numa forma que suporte a manipulação automatizada (modelagem, execução) por um sistema de gerenciamento de workflow. A definição de processo consiste da rede das atividades e seus relacionamentos, critérios para iniciar ou terminar o processo e informação sobre participantes, aplicações e dados associados. Pode incluir a definição dos aspectos manuais e dos aspectos automatizáveis do processo.
Definição de Workflow (Workflow Definition): é a parte da Definição de Processo que inclui só os seus aspetos automatizáveis. Se se faz uma diferenciação entre uma Definição de Processo e as atividades dentro dela que são automatizáveis, o termo Definição de Workflow é usado.
Atividade(Activity): a descrição de um elemento de trabalho que forma um passo lógico dentro de um processo. Uma atividade dentro do workflow requer recursos humanos ou de máquina para suportar sua execução; se requerer recursos humanos ela é alocada a um par-

ticipante do workflow. Uma atividade, como unidade mínima de trabalho, é programada por uma máquina de workflow, durante a execução do processo, embora possa resultar em vários ítens de trabalho, alocados a um participante do workflow.
Participante do Workflow (Workflow Participant): é o recurso que realiza o trabalho representado por uma instância de uma atividade. Esse trabalho geralmente manifesta-se como um ou vários ítens de trabalho, que são alocados ao participante do workflow, via uma lista de trabalho.
Item de trabalho: a representação do trabalho a ser executado por um participante do workflow, no contexto de uma atividade, dentro de uma instância de processo. Um item de trabalho não forma um passo lógico dentro do processo e é a forma de associar uma instância de atividade com um participante de workflow.
Lista de trabalho: uma lista de itens de trabalho, associada com um dado participante do workflow (ou em alguns casos com um grupo de participantes do workflow que compartilham a mesma lista de trabalho).
Sistema de Gerenciamento de Workflow (Workflow Management System): o sistema de gerenciamento de workflow é composto por um conjunto de produtos de software que permitem definir o processo de workflow, analisá-lo, executá-lo automaticamente e monitorá-lo. O sistema basicamente é formado por dois módulos, um para ser usado no momento de projetar a representação computadorizada da lógica do workflow, e outro usado no momento da execução do sistema. O módulo usado no momento do projeto consiste na Ferramenta para Definir o Processo. O módulo usado no momento de execução é conhecido como o Serviço de Execução do Workflow e provê o ambiente computacional para a execução do processo de workflow. Este serviço de software pode ser composto por máquinas de workflow para criar, administrar e executar instâncias particulares de workflow. As aplicações interagem com este serviço via a WAPI
Ferramentas para Definir o Processo (Process Definition Tools): ferramentas usadas para capturar a lógica do processo de negócio, numa notação de alto nível.
Serviço de Execução do Workflow (Workflow Enactment Service): é o software servidor do workflow e nervo central do sistema de workflow. É responsável pela administração do processo, distribuição e invocação das atividades. Para a execução de instâncias específicas de workflow usam-se máquinas de workflow.
Máquina de Workflow (Workflow Engine): este serviço de software cria, administra e executa instâncias específicas de workflow. As máquinas de workflow fazem parte do Serviço de Execução do Workflow e são administradas por ele.
Aplicações Cliente do Workflow (Workflow Client Applications): são uma aplicações com interface gráfica para o usuário ver e administrar no servidor de workflow (Serviço de Execução do Workflow) o conteúdo de sua lista de trabalho e interagir com seus itens de trabalho no servidor.

WAPI (Workflow Application Programming Interface): as interfaces para que as aplicações de workflow e ferramentas possam interagir com o sistema de gerenciamento do workflow. Essas especificações visam a interoperabilidade entre os diferentes componentes de um sistema de gerenciamento de workflow e as aplicações.

Em resumo, os esforços de padronização da WfMC em workflow foram orientados à criação de especificações de interoperabilidade, que podem classificar-se em duas categorias:

- especificações para a modelagem e definição de processos de workflow e
- especificações para interoperabilidade em tempo de execução

Especificações para a modelagem e definição de processos de workflow

O trabalho da WfMC, na área de modelagem e definição de processos de workflow, centrou-se na especificação de um metamodelo para a definição de processos de workflow [1.5], o qual será apresentado no capítulo 2. Acompanhando o metamodelo citado, definiu-se uma representação textual dele, conhecida como 'A Linguagem de Definição de Processos de Workflow da WfMC' (The WfMC Workflow Process Definition Language) [1.5] e que corresponde à especificação da interface 1 ilustrada na Figura 1.1

Especificações para interoperabilidade em tempo de execução

O trabalho da WfMC na área de padrões de execução corresponde:

- à especificação do conjunto de APIs para a interação entre as aplicações (Aplicações Cliente do Workflow e Aplicações Invocadas) e o Serviço de Execução do Workflow, conhecida como "WfMC API Standard" [1.6] e que corresponde à especificação das interfaces 2 e 3 da Figura 1.1.
- à especificação de interoperabilidade entre servidores de workflow, conhecida como "WfMC Workflow Interoperability Standard" [1.7] e que corresponde à interface 4 da Figura 1.1 e
- à especificação que define a informação que necessita ser registrada como conseqüência das mudanças de estado que acontecem durante a execução do workflow, conhecida como "WfMC Audit Data Specification" [1.8] e que corresponde à especificação da interface 5 na Figura 1.1.

Em síntese, o Modelo de Referência da WfMC define uma arquitetura cliente –servidor, na qual o Serviço de Execução do Workflow é monolítico e centraliza todos os serviços do workflow, como o gerenciamento do processo, a distribuição das atividades, o gerenciamento da lista de trabalho, e o serviço de diretório, não definindo as interfaces internas desses serviços. O anterior reflete o fato de que os sistemas tradicionais de workflow foram projetados para um ambiente homogêneo, no qual todos os workflows seriam especificados num único modelo de especificação de workflow proprietário, executado num ambiente de

execução proprietário, e utilizando recursos humanos e computacionais registrados num único serviço de diretório.

O atual ambiente de redes globais e conectividade entre várias empresas obrigou o surgimento de novos padrões que enfocam os problemas de execução de workflows descentralizados e distribuídos, via componentes, que interoperam através da infra estrutura de rede global. Os esforços nesse sentido têm sido liderados pelo "Grupo para Gerenciamento de Objetos" (Object Management Group, OMG) [1.10] e serão brevemente apresentados na Seção 1.2.

1.2 A 'OMG' E OS ESFORÇOS DE PADRONIZAÇÃO RELACIONADOS COM WORKFLOW

Os esforços de padronização em workflow na OMG foram orientados à criação de especificações de interoperabilidade que podem também classificar- se em duas categorias:

- especificações para interoperabilidade em tempo de execução
- especificações para modelagem e descrição de processos de workflow

O trabalho da OMG na área de padrões de execução

A primeira tarefa abordada pela OMG com respeito a padrões de workflow foi a relacionada com os padrões de execução do workflow. A Facilidade de Workflow (Workflow Management Facility) [1.11] descreve o comportamento associado com a execução de um sistema de workflow. Esse padrão adapta os padrões de execução da WfMC a um ambiente de execução de objetos. Ele define um sistema de interfaces para implementar aplicações distribuídas de workflow que visa a interoperabilidade dos componentes do workflow, o monitoramento, execução e associação das componentes dele com os recursos necessários para sua execução.

Muitos aspectos da execução do workflow dependem da definição do workflow, a qual não foi considerada no padrão de execução com o objetivo de limitar o alcance dele. O processo correspondente à adoção de um padrão para a definição do workflow, por parte da OMG, está atualmente em andamento e será exposto mais adiante neste capítulo.

Outro aspecto de execução que também depende da definição do workflow é a designação dos recursos. A Facilidade de Workflow não define um modelo de recursos embora especifique a interface (WfResource interface) como um lugar para as entidades desse modelo. No entanto, está em andamento, de forma separada, o processo para expandir os aspectos da modelagem dos recursos através de uma Chamada de Propostas para uma Facilidade de Designação de Recursos (Resource Assignment Facility).

O trabalho da OMG na área de modelagem e definição de processos de workflow

Dado que a OMG havia definido a Linguagem de Modelagem Unificada (UML- Unified Modeling Language) [1.13] como padrão para modelagem, o trabalho específico de mode-

lagem da definição de processo de workflow está circunscrito a ela e iniciou-se com a publicação de uma Chamada de Propostas, solicitando extensões à UML para expressar Definições de Processo de Workflow (UML Extensions for Workflow Process Definition, Request for Proposals) [1.19]. A Requisição solicita, em particular, propostas para um metamodelo e/ou perfil que estenda a UML para definir processos de workflow. O processo de submissão de respostas está ainda em andamento (Junho de 2002).

Como um dos objetivos desta tese é a apresentação de um metamodelo unificado, expressado como um perfil de UML, e com o objetivo de posicionar esse perfil dentro do marco geral de padronização da OMG, as seguintes seções esclarecem primeiro o conceito de metamodelo (seção 1.2.1) e perfil de UML (seção 1.2.2). Depois é apresentada a arquitetura de metamodelos da OMG (seção 1.2.3) e nela é posicionado o perfil em questão.

1.2.1 O Conceito de Metamodelo

Em geral, os modelos são usados para facilitar o entendimento dos sistemas reais e, como tal, tentam representar a realidade de uma forma simplificada. Mas os modelos não são uma representação direta do sistema de nosso interesse mas da *interpretação* subjetiva dessa parte do mundo real que é de nosso interesse. Com o fim de ilustrar estes conceitos e os relacionamentos entre eles (ver Figura 1.2, adaptada de [1.29]) vamos chamar essa ***interpretação*** subjetiva do mundo real, e que é de nosso interesse, como Sistema Objeto de interesse (So) e uma ***representação*** dele como o seu modelo (M1).

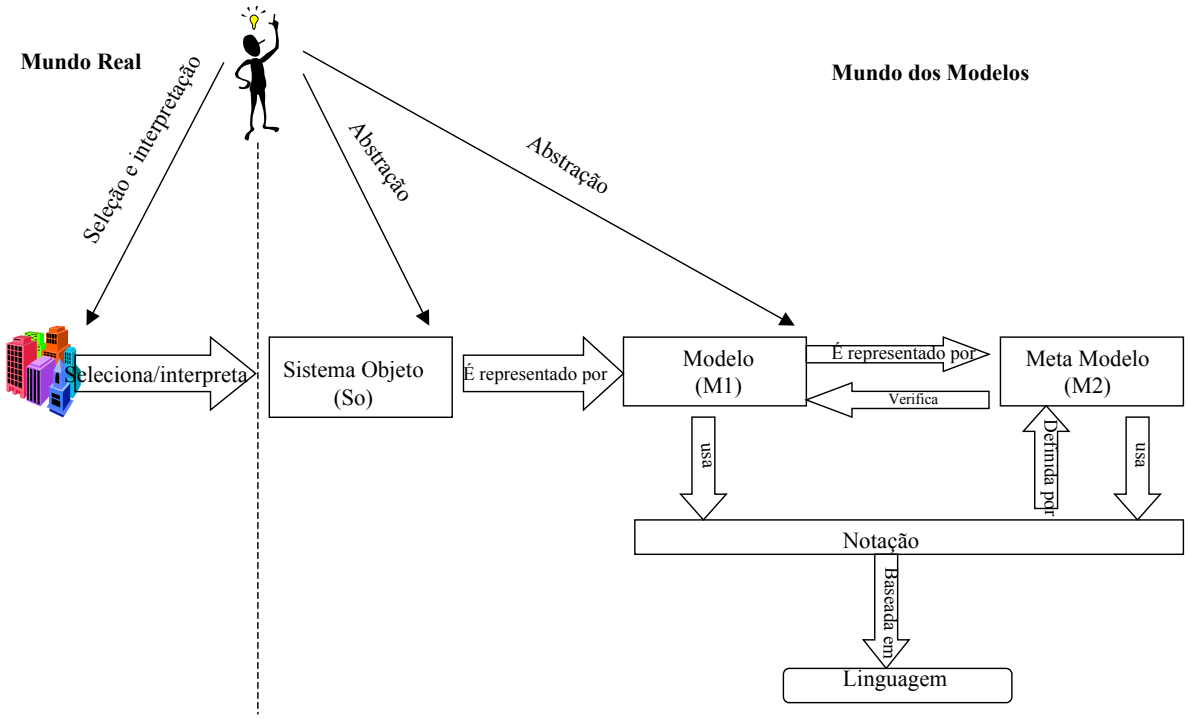


Figura 1.2. Relações entre Realidade, Interpretação, Modelo e Metamodelo [1.29].

Se, por sua vez, esse modelo (M1) passa a ser um novo Sistema Objeto de interesse a ser ***representado*** por um outro modelo (M2) então esse modelo (M2) passa a ser o chamado **metamodelo** do modelo M1.

No contexto desse grau de abstração, o metamodelo (M2), que é o modelo do modelo (M1), descreve os elementos básicos de modelagem usados para criar o modelo (M1). O metamodelo (M2) define os possíveis elementos de modelagem a serem usados na construção do modelo (M1), seus relacionamentos, semântica, regras para usá-los e especializá-los. É necessária, tanto uma linguagem (uma notação e sintaxes) para representar o modelo M1, quanto uma para representar o metamodelo (M2).

Todo metamodelo, para ser descrito, necessita de outro metamodelo que defina os elementos de modelagem a serem usados na sua construção . O metamodelo no qual está baseado um metamodelo pode ser da mesma classe (por exemplo o metamodelo de UML é definido usando elementos de UML). Se as semelhanças entre vários metamodelos podem ser consolidadas num outro modelo universal, que tenha um alto grau de qualidade semântica para poder expressar cada um deles, podemos falar de um metamodelo de Referência ou MetaMetamodelo, como é o caso do MetaMetaModelo de Referência MOF[5] na arquitetura de metamodelos de quatro níveis da OMG (ver Figura 1.3 na seção 1.2.3, 'O Conceito da Arquitetura de Metamodelos na OMG').

1.2.2 O conceito de Perfil

Para evitar que a Linguagem de Modelagem Unificada UML fosse muito complexa, quando existissem conceitos particulares num domínio particular de aplicação que não estivessem definido na UML, mas que fossem suficientemente importantes para serem definidos na linguagem de modelagem, eles seriam criados usando o mecanismo de extensão que UML prove para definir os novos elementos a partir de elementos de modelagem[6] já existentes. Isso permite ao modelador criar modelos mais claros e precisos.

Dentro do contexto da UML, o Perfil [1.13], [1.24] é uma especificação que especializa um metamodelo padrão usando os mecanismos de extensão que o metamodelo de UML fornece: "Tagged Values"[7], "Constraints"[8] e "Stereotypes"[9]. Um perfil visa adaptar o me-

[5] MOF (Meta Object Facility): especifica as interfaces das funcionalidades de um repositório de tipos de metadados para diferentes sistemas tipificados e é capaz de manter os relacionamentos entre os tipos dos diferentes sistemas tipificados. MOF prescreve o uso de tipos abstratos (metaclasses), associações abstratas (meta associações), multiplicidade e "roles". O padrão MOF provê meios para estender os metamodelos através da definição de novas metaclasses. Todos os metamodelos na arquitetura de modelos da OMG devem poder ser descritos com base nos elementos definidos no MetaMetaModelo MOF.

[6] Um elemento de modelagem é uma abstração, suas especializações são os conceitos particulares (artefatos) como: classe, nó, transição etc.

[7] Tagged Value: é um par constituído por uma propriedade e seu respectivo valor que permite que informação adicional seja agregada a um elemento de modelagem. A interpretação da propriedade é determinada pelo usuário. Ex. pré_condições e pós_condições agregadas às operações de uma classe.

tamodelo padrão definindo novos conceitos específicos num domínio especifico de aplicação. Um perfil de UML faz uma ou mais das seguintes funções.

- Identificar um subconjunto do metamodelo de UML (poderia ser também todo o metamodelo).
- Especificar novas "regras bem formadas" além das especificadas no subconjunto do metamodelo de UML escolhido. "Regras bem formadas" é uma expressão usada na especificação do metamodelo de UML para descrever o conjunto das restrições da UML na linguagem OCL (Object Constraint Language)[10], as quais contribuem para a definição dos elementos do metamodelo.
- Especificar novos "elementos padrão" além dos especificados no sub_conjunto do metamodelo de UML escolhido. "Elemento padrão" é um termo usado na especificação de metamodelo da UML para descrever instâncias padrão de um "stereotype", "tagged value" ou "constraint" do UML.
- Especificar uma nova semântica, expressa em linguagem natural, além daquela especificada no subconjunto do metamodelo de UML identificado.
- Especificar os elementos de modelagem comuns (instâncias dos elementos da UML) expressando-os em termos do perfil.

1.2.3 O Conceito da Arquitetura de Metamodelos na OMG

Com o objetivo de ter um repositório comum para os diferentes tipos de metamodelos padrão que fazem parte da arquitetura da OMG, esta padronizou uma arquitetura com 4 níveis, mostrada na Figura 1.3. O nível de MetaMetaModelos (M3, na Figura 1.3) é definido pela Facilidade de Meta Objetos (MOF- Meta Object Facility) [1.23], que é a base sobre a qual os diferentes metamodelos padrão devem poder ser definidos. O exemplo mais notável de metamodelo padrão nessa arquitetura é o metamodelo de UML. Existem outros metamodelos padrão, como o metamodelo de interfaces de execução da Facilidade de Workflow [1.11]. Espera-se padronizar outras áreas provendo metamodelos padrão específicos.

---

[8] Constraint: é uma restrição que permite que uma nova semântica seja especificada para um elemento de modelagem através de uma restrição que é agregada e que o elemento de modelagem deve obedecer.

[9] Stereotype: este mecanismo adapta a semântica de um elemento de modelagem sem troca desse elemento. De tal maneira que o 'stereotype' pode ser visto como um novo elemento. Isto permite ao modelador criar modelos mais claros e precisos. Pode especificar novos "Tagged Values" e novas "Constraints". Um elemento de modelagem pode ter agregado nenhum ou um estereótipo.

[10] Object Constraint Language: na modelagem orientada a objetos, um modelo gráfico como, por exemplo, um modelo de classes, não é suficiente para sua especificação precisa e não ambígua. É necessário descrever restrições adicionais acerca dos objetos do modelo. OCL é uma linguagem formal que faz parte de UML e que permite expressar de maneira não ambígua restrições que esclarecem o significado semântico dos elementos da modelagem.

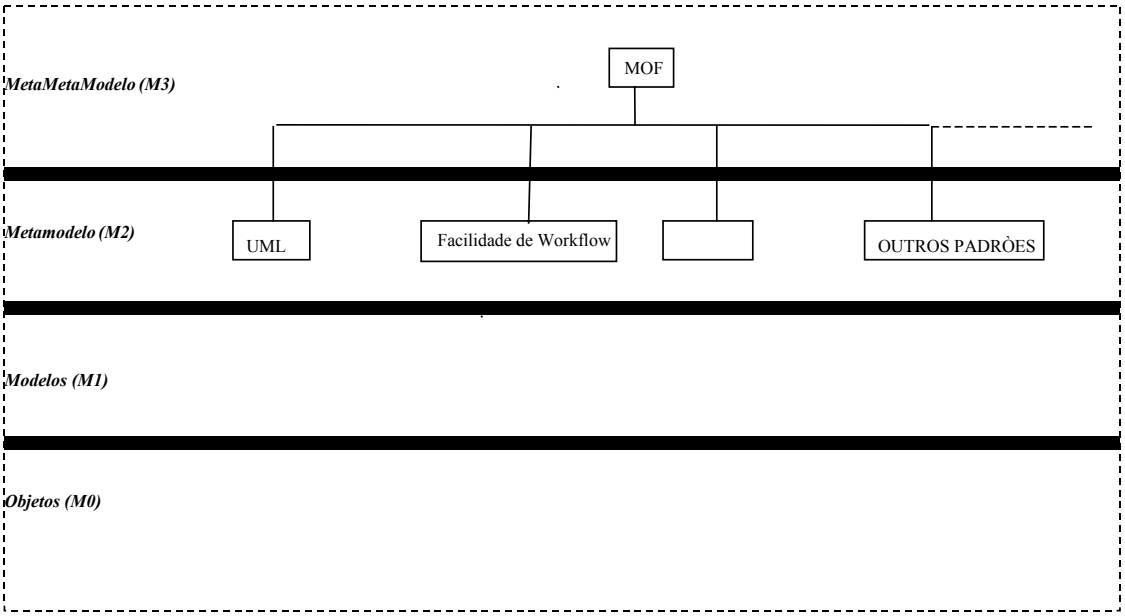


Figura 1.3 A Arquitetura de metamodelos de quatro níveis na OMG

Na arquitetura de metamodelos em quatro níveis da OMG, cada metamodelo padrão (nível M2) define os elementos de modelagem necessários para criar modelos do nível M1 numa área específica. Os metamodelos do nível M2 (seus elementos, relacionamentos, semântica, regras de uso) são usados para definir os modelos do nível M1. No nível M0, encontram-se as instâncias específicas dos modelos definidos no nível M1 anterior. O MetaMetaModelo MOF do nível M3 representa o metamodelo de Referência mencionado no último parágrafo da seção 1.2.1. Todos os demais metamodelos devem ser compatíveis com ele. Em resumo, nesta arquitetura cada modelo é uma instância do modelo do nível anterior.

As áreas padrão cobertas por um metamodelo podem ser decompostas em domínios de aplicação específicos, que necessitam de refinamentos específicos do metamodelo padronizado. Em UML o perfil (ver seção 1.2.2) é a forma de adaptar o metamodelo padrão definindo novos conceitos específicos num domínio específico de aplicação. Tal é o caso com o Perfil EDOC [1.20], sob a área de UML, que provê uma semântica dedicada ao domínio dos sistemas de computação empresarial (ver Figura 1.4).

Na Figura 1.4 é ilustrada a localização do perfil EDOC, do Perfil do metamodelo unificado e de outros metamodelos padrão do nível M2 que potencialmente poderiam ser usados para criar o modelo de uma aplicação baseada em workflow (modelo no nível M1).

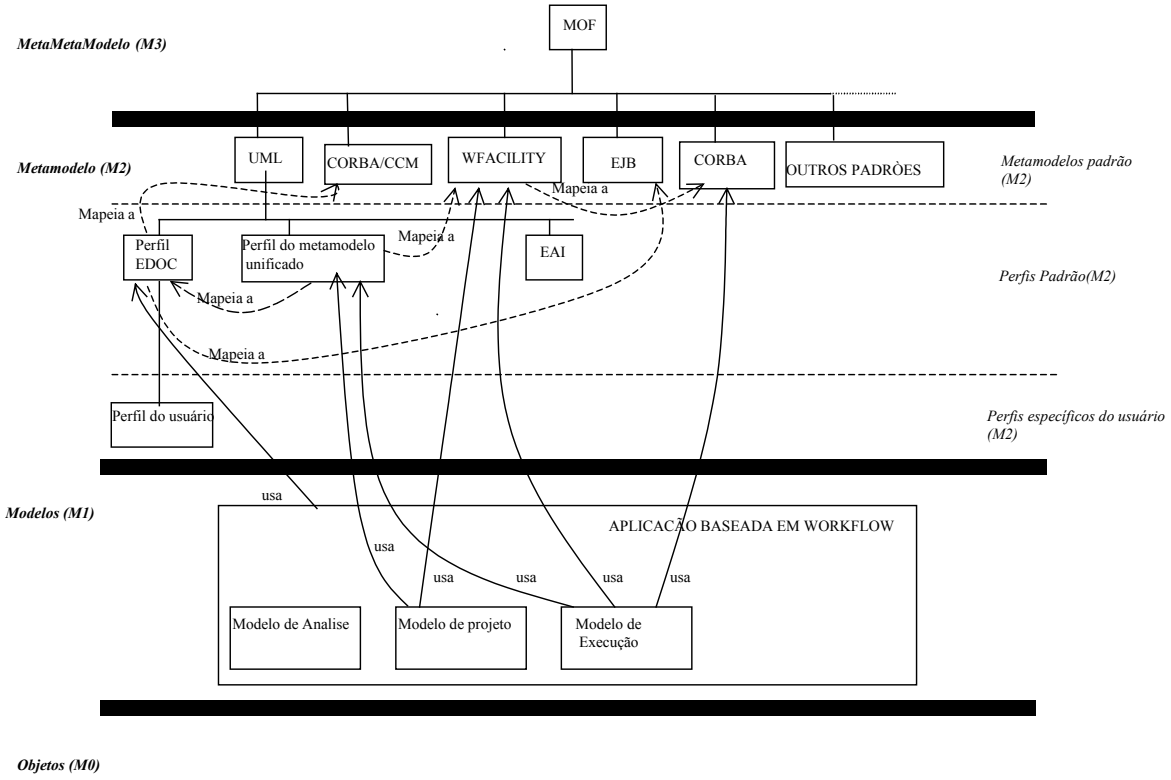


Figura 1.4 O metamodelo unificado na arquitetura de metamodelos em quatro níveis da OMG

O software de suporte para automatizar os processos de negócio de uma organização se traduz no desenho de uma aplicação de workflow (nível M1 na Figura 1.4) e sua posterior implementação numa plataforma de execução específica. Toda aplicação de workflow tem no mínimo um modelo que captura a lógica do processo de negócio, Definição de Workflow em termos da WfMC, (ver Tabela 1.1) e de um Serviço de Execução de Workflow para executá-la.

A Figura 1.4 está mostrando, através do relacionamento 'usa' entre o 'Modelo de projeto' e o Perfil EDOC, que a especificação do projeto da aplicação de workflow (nível M1) faz uso do perfil EDOC (nível M2). Ao mesmo tempo a especificação do modelo de execução (nível M1), que interpreta e executa a lógica do processo de workflow, 'usa' o perfil do metamodelo unificado que serviu para especificá-la. A coerência, no sentido de que a implementação da aplicação com workflow, modelada usando o perfil EDOC, seja consistente com a lógica do processo de workflow, vai estar garantida mediante a formulação (no Capítulo 3) de um metamodelo unificado consistente com EDOC e pela existência de um mapeamento entre os dois perfis. Em particular o perfil do metamodelo unificado mapeia ao perfil EDOC (ver Figura 1.4).

Outros relacionamentos ilustrados na Figura 1.4 podem ser melhor explicados sob o enfoque adotado na OMG, como padrão para especificar sistemas de software, conhecido como a Arquitetura Orientada por Modelos (MDA-Model Driven Arquitecture). Este enfoque está baseado nas diferentes perspectivas (viewpoints) do Modelo de Referência para Processamento Distribuído Aberto (Reference Model of Open Distributed Processing, RM-ODP) [1.25], [1.26] que busca especificar, por separado, os diferentes aspectos do sistema, particionando-o em cinco especificações : (i) empresarial, (ii) computacional, (iii) de informação, (iv) de engenharia, e (v) tecnológica. Ao mesmo tempo que o enfoque enfatiza a formulação de modelos que sejam independentes da plataforma de implementação, conhecidos como modelos PIM, (Plataform Independent Models) e seu posterior mapeamento às diferentes tecnologias de implementação.

Sob este enfoque os relacionamentos 'mapeia' entre o perfil EDOC e Corba/CCM (modelo de Componentes de CORBA) [1.27] e entre EDOC e EJB (Enterprise JavaBeans[1.28]) ilustram o mapeamento entre o modelo gráfico da aplicação de workflow independente da plataforma (PIM) e potenciais alternativas tecnológicas de implementação (Corba/CCM e/ou EJB). Ao mesmo tempo, está ilustrado através do relacionamento 'usa' entre os modelos de projeto de execução e o metamodelo WfFacility, a possibilidade de que o metamodelo padrão de interfaces de execução de workflow da OMG [1.11] seja utilizado no desenho e execução de uma aplicação com workflow implementada na plataforma CORBA.

Sob o enfoque específico das perspectivas do Modelo de Referência para Processamento Distribuído Aberto (RM-ODP) a definição de processo de workflow a ser modelada usando o perfil do metamodelo unificado corresponde à modelagem da perspectiva empresarial[11]. As outras perspectivas, (ii) computacional[12], (iii) de informação[13] e de (iv) de engenharia[14]

---

[11] A especificação empresarial descreve como o sistema suporta os processos de negócios em termos :

- dos objetivos dos processos de negócio que vão ser suportados pelo sistema,
- dos passos desses processos e os relacionamentos entre os passos,
- das regras de negócio que se aplicam a esses passos,
- dos papéis ('roles') das entidades do sistema que suportam o processo de negócio.

[12] A especificação da perspectiva computacional descreve a funcionalidade do sistema em termos de um conjunto de objetos computacionais que interagem e que correspondem a uma partição apropriada para distribuição. Descreve os principais componentes funcionais do sistema, o que cada um faz e como se relacionam entre si, sem prescrever se estão ou não distribuídos em diferentes plataformas. Esta especificação estabelece a capacidade de ser distribuído mas não descreve como a distribuição pode ser implementada. A especificação define:

- os objetos computacionais que exercem algum papel 'role' funcional no sistema e que podem ser descritos em termos das interfaces que fornece e que usa ,
- as interfaces que o objeto computacional usa para interagir,
- as estruturas de colaboração entre os objetos computacionais.

[13] Esta especificação concentra-se na semântica e no processamento da informação envolvida com aquelas partes do processo de negócio suportadas pelo sistema. É um modelo de objetos que especifica a informação que representa os artefatos e os recursos sobre os quais o sistema age.

[14] Esta especificação é centrada na infra-estrutura que suporta a operação dos objetos computacionais, em termos dos mecanismos que lhes permitem interagir e que provêm a requerida transparência de distribuição.

são modeladas usando os diferentes sub- perfis que fazem parte do Perfil EDOC e a perspectiva (v) tecnológica[15] é modelada usando os correspondentes mapeamentos para as tecnologias específicas, como EJB e CORBA, com o modelo de componentes, .Net com DCOM etc.

A seguinte seção introduz a estrutura geral da tese.

1.3 ESTRUTURA DA TESE

No capítulo 2, são apresentados um resumo e a comparação dos modelos que foram usados como referência para guiar a proposta do metamodelo unificado de processos de workflow (associados com a automação de processos de negócio) e dos sistemas de software que dão suporte a eles (aplicações com workflow).

O capítulo 3 apresenta o metamodelo unificado desenvolvido e propõe uma representação textual plausível, correspondente à modelagem baseada no metamodelo unificado. A notação textual proposta é um enfoque plausível para transformar uma representação gráfica de uma definição de processo de workflow, numa representação computacionalmente interpretável por uma máquina de workflow

O capítulo 4 apresenta alguns aspectos da implementação, derivada e apoiada nos resultados teóricos mencionados no capítulo 3, em particular ilustrando-se os resultados da integração e execução de uma máquina de workflow numa plataforma que fornece funcionalidades básicas para o desenvolvimento de um mercado de serviços (a plataforma Platin) [1.21]. O protótipo de máquina de workflow, implementado na linguagem Java e CORBA, interpreta definições de processo de workflow expressas na notação textual proposta no capítulo 3.

O capítulo 5, apresenta algumas conclusões e relaciona alguns aspectos cuja abordagem seria de interesse em trabalhos futuros.

A tese finaliza com as Referências Bibliográficas de todos os capítulos e um Apêndice listando os artigos gerados durante o desenvolvimento do trabalho.

Modela a implementação da especificação computacional dentro de uma infra-estrutura particular. As características da infra-estrutura definida devem ser consistentes com as especificações Empresarial, Computacional e de Informação

[15] Esta especificação mapeia os componentes da especificação da perspectiva de engenharia na realização específica desses componentes. Identifica os pontos físicos de referência

1.4 REFERÊNCIAS DO CAPÍTULO

[1.1] Knutilla, A.; Schlenoff, C.; Ray, S.; Polyak, S. T.; Tate, A.; Cheah S.C.; Anderson,R. C. "Process Specification Language: An Analysis of Existing Representations".[http://www.mel.nist.gov/psl], último acesso (10/10/2001).

[1.2] The Workflow Management Coalition, WfMC.[http://www.wfmc.org]

[1.3] Workflow Management Coalition. Terminology and Glossary. Document Number WFMC-TC-1011, June 1996.

[1.4] Workflow Reference Model. Document Number WfMC-TC-1003 v1.1 (Jan 95).
[http://www.wfmc.org]

[1.5] Interface 1 – Process Definition Interchange- Process Model, v 1.1WfMC-TC-1016-P, Workflow Management Coalition, October 29, 1999; [http://www.wfmc.org]

[1.6] Workflow Client API Specifications (WAPI); WfMC-TC-1002, v2.0 (Jul 98).
[http://www.wfmc.org]

[1.7] Workflow Interoperability- Abstract Specifications; WfMC-TC-1012, v2.0 (Dec 99). [http://www.wfmc.org]

[1.8] Workflow Audit Data Specification; WfMC-TC-1015, v1.1 (Sep 98).
[http://www.wfmc.org]

[1.9] W. M. P van der Aalst, K. M. van Hee and G. J. Houben. "Modeling and Analysing Workflow using a Petri-net based Approach". Proc. of the 2nd Workshop on Computer Supported Cooperative Work, Petri Nets and Related Formalisms. [http://wwwis.win.tue.nl/~wsinwa/wfm-adv.ps]
W. M. P van der Aalst. "The Application of Petri Nets to workflow Management"
[http://wwwis.win.tue.nl/~wsinwa/jcsc/jcsc.html]

[1.10] 'The Object Management Group (OMG)' , http://www.omg.org.

[1.11] Juergen Boldt. Workflow Management Facility Specification (WMF), v1.2, OMG Document Number: formal/00-05-02; 2000
[http://ftp.omg.org/pub/docs/formal/00-05-02.pdf]

[1.12] Richard Soley (ed.). "Object Management Architecture Guide", Third Edition, Wiley, June 1995. OMG Document ab/97-05-5; 1997.
[http://ftp.omg.org/pub/docs/ab/97-05-5.pdf]

[1.13] Ms. Linda Heaton. "Unified Modeling Language Specification v1.4 "; (2001) OMG Document Number: formal/01-09-67 a formal/01-09-80 [http://ftp.omg.org/pub/docs/formal/01-09-67.pdf]

[1.14] "UML Profile for Enterprise Distributed Object Computing", –Request for Proposal, OMG Document Number: ad/99-03-10; 1999.
[http://ftp.omg.org/pub/docs/ad/99-03-10.pdf]

[1.15] A.P. Sheth, W. van der Aalst, and I.B. Arpinar, "Process Driving the Networked Economy", IEEE Concurrency, Vol. 7, No. 3, July-September 1999, [http://dlib.computer.org/pd/books/pd1999/pdf/p3018.pdf]

[1.16] Arpinar S; Dogac A; Tatbul N. "An Open Electronic Marketplace through Agent-based Workflows:MOPPET".
[http;//www.srdc.metu.edutr/srdc_publications.html]

[1.17] A. Dogac, et. al., "A Workflow-based Electronic Marketplace on the Web",
[http://www.acm.org/sigmod/sigmod_record/issues/9812/SPECIAL/dogac.pdf.gz]

[1.18] Bussler Christoph. "Enterprise-Wide Workflow Management.". IEEE Concurrency, Vol. 7, No. 3, July-September 1999.[http://dlib.computer.org/pd/books/pd1999/pdf/p3032.pdf]

[1.19] "UML Extensions for Workflow Process Definition" –Request for Proposal, OMG Document Number: bom/2000-12-11; 2000.
[http://ftp.omg.org/pub/docs/bom/00-12-11.pdf]

[1.20] UML Profile for Enterprise Distributed Object Computing-EDOC Draft Adopted Specification OMG Document Number: ptc/2001-12-04; 2001.
[ftp://ftp.omg.org/pub/docs/ptc/01-12-04.pdf]

[1.21] Eckert K, Körner E, Schoo P. "Introduction to Middleware Platform", PLATIN Platform Documentation version 1.0, 2001. FOKUS-Research Institute for Open Communication System, Berlin. [http://www.fokus.gmd.de]

[1.22] Schenk, M. "OMG White Paper Open Service Marketplace", OMG Document, May 2000.[ftp://ftp.omg.org/pub/docs/telecom/00-05-02.pdf]

[1.23] Meta Object Facility (MOF) v1.3.1, OMG Document Number: formal/01-11-02; 2001.
[ftp://ftp.omg.org/pub/docs/formal/01-11-02.pdf]

[1.24] "White Paper on the Profile Mechanism", OMG Document Number: ad/99-04-07; 1999.
[ftp://ftp.omg.org/pub/docs/ad/99-04-07.pdf]

[1.25] ISO/IEC & ITU-T: Information technology – Open Distributed Processing – Reference Model - Architecture – ITU-T Recommendation X.903 | ISO/IEC 10746-3

[1.26] ISO/IEC & ITU-T: Information technology – Open Distributed Processing – Reference Model - Enterprise Language – ITU-T Recommendation X.911 | ISO/IEC 15414, (2001), OMG Document : ISO-std/01-01-01; 2001.
[ftp://ftp.omg.org/pub/docs/ISO-std/01-01-01.pdf]

[1.27] CORBA Component Model, OMG Document Number: ptc/01-11-02; 2001.
[ftp://ftp.omg.org/pub/docs/ ptc/01-11-02.pdf]

[1.28] Enterprise JavaBeans Specification. Sun Microsystems Inc.
[http://www.javasoft.com/products/ejb/newspec.html]

[1.29] Michael zur Muhlen. "Evaluation of Workflow Management Systems Using Meta Models". Proceedings of the 32nd Hawaii Internacional Conference on System Sciences, 1999, (HICSS-1999).

CAPÍTULO 2

MODELOS DE REFERÊNCIA E COMPARAÇÕES

Este capítulo apresenta um resumo dos modelos que serão usados como referência para guiar a proposta de um metamodelo unificado para modelar processos de workflow e seu software de suporte associado, o qual será apresentado no capítulo 3.

Os modelos serão apresentados na ordem cronológica em que foram publicados. Inicialmente serão apresentados os conceitos relacionados com os metamodelos de workflow da WfMC (Workflow Management Coalition) e da Nortel (Northern Telecom) extraídos de [2.1] e [2.2] respectivamente. Posteriormente serão brevemente apresentados os conceitos necessários para a modelagem de processos de negócio segundo o perfil padrão de UML para especificar Sistemas de Computação de Objetos Empresariais Distribuídos (UML Profile for Enterprise Distributed Object Computing, EDOC) [2.3]. De aqui em adiante este perfil, quando for mencionado, será citado simplesmente como Perfil EDOC.

O primeiro metamodelo citado como metamodelo de workflow da WfMC foi o primeiro padrão para definição de processos de workflow aceito pela comunidade de workflow. O trabalho da WfMC, nesse sentido, se iniciou em 1995 com a análise de várias linguagens de definição de processos de vários fornecedores de produtos de workflow, até chegar a um metamodelo mínimo de processos de workflow. Acompanhando o metamodelo citado definiu uma representação textual dele, conhecido como 'A Linguagem de Definição de Processos de Workflow da WfMC' (The WfMC Workflow Process Definition Language) [2.1], e que corresponde à especificação da interface 1 do Modelo de Referência de Workflow ilustrado na Figura 1.1 do Capítulo 1. Embora não seja ele um metamodelo orientado a objetos, continua sendo, por sua simplicidade e clareza, o modelo de workflow mais amplamente conhecido e divulgado.

O segundo modelo de workflow, acima citado como modelo da Nortel, foi apresentado inicialmente como resposta à Chamada de Propostas [2.4] da OMG para padronizar um Serviço de Workflow (Workflow Facility) em sua Arquitetura para Gerenciamento de Objetos (OMA- Object Management Architecture). Embora essa Requisição visasse especificar só as interfaces de execução de um Serviço de Workflow, a proposta da Nortel incluiu também um metamodelo básico de processos de workflow, coerente com as interfaces de execução. E, embora esta proposta não tenha sido a que finalmente foi adotada, ela é arquitetonicamente neutra, não favorecendo nenhuma modelagem empresarial específica ou nenhuma metodologia para definição de processos de workflow, já que foi desenvolvida baseada num metamodelo de workflow mínimo (básico). A proposta adotada conhecida como a JFlow Facility [2.13] apresentou só a especificação das interfaces de execução de um Serviço de Workflow mas dado que não incluiu nenhum metamodelo para representar definições de processo de workflow não foi considerada para guiar a proposta do metamodelo unificado.

Alem disso, a proposta da Nortel apresenta na formalização de seu metamodelo elementos para definir um processo de workflow, que são coerentes com os conceitos usados no Perfil UML para Processos de Negócio [2.1]. Embora na proposta da Nortel os metaconceitos não estejam definidos em UML eles são compatíveis com o modelo de MOF.

No entanto, o perfil EDOC, recentemente aprovado (Dezembro de 2001), como padrão em qualidade de 'draft" pelo OMG [2.3], para especificar os *aspectos do desenho* dos sistemas empresariais distribuídos, tem em consideração que o sucesso da implementação de tais sistemas requer que a operação deles esteja diretamente relacionada com os processos de negócio que eles suportam.

As razões citadas levaram à escolha inicial destes dois modelos de workflow, para uma comparação com o metamodelo de processos de negócio (Perfil EDOC), da qual virão a deduzir os principais elementos da proposta de um metamodelo unificado para modelar processos de workflow e seu software de suporte.

A seções 2.1 e 2.2 apresentam os metamodelos de workflow da WfMC e da Nortel respectivamente. A seção 2.3 apresenta um resumo do perfil EDOC. Finalmente a seção 2.4 apresenta algumas comparações feitas entre esses metamodelos. A comparação realizada está inspirada na metodologia proposta por Muhler em [2.5] para comparar sistemas de workflow, baseada em seus metamodelos, a qual será introduzida na seção 2.4.1 deste capítulo.

Para facilitar a diferenciação do contexto de proveniência os elementos que fazem parte do modelo de WfMC serão apresentados com letras sublinhadas, em *itálico sublinhados* os que provêm do modelo da Nortel e em **negrito** os conceitos extraídos do perfil EDOC. Para facilitar a referência com outros documentos, a descrição de cada meta entidade, pela primeira vez citada, estará acompanhada pela respectiva transcrição ao Inglês tal como aparece na fonte original.

2.1 O METAMODELO DE WORKFLOW DA WfMC

Para proporcionar um método comum para acessar e descrever definições de workflow a WfMC introduziu um metamodelo de dados para definições de processo de workflow. Esse metamodelo identifica as entidades normalmente usadas dentro de uma definição de processos e um conjunto de atributos descreve as características de cada uma das entidades. O metamodelo descreve as principais entidades, seus relacionamentos e atributos. A Figura 2.1 mostra um diagrama de classes em UML do metamodelo para definição de processos de workflow da WfMC, adaptado a partir do apresentado em [2.1].

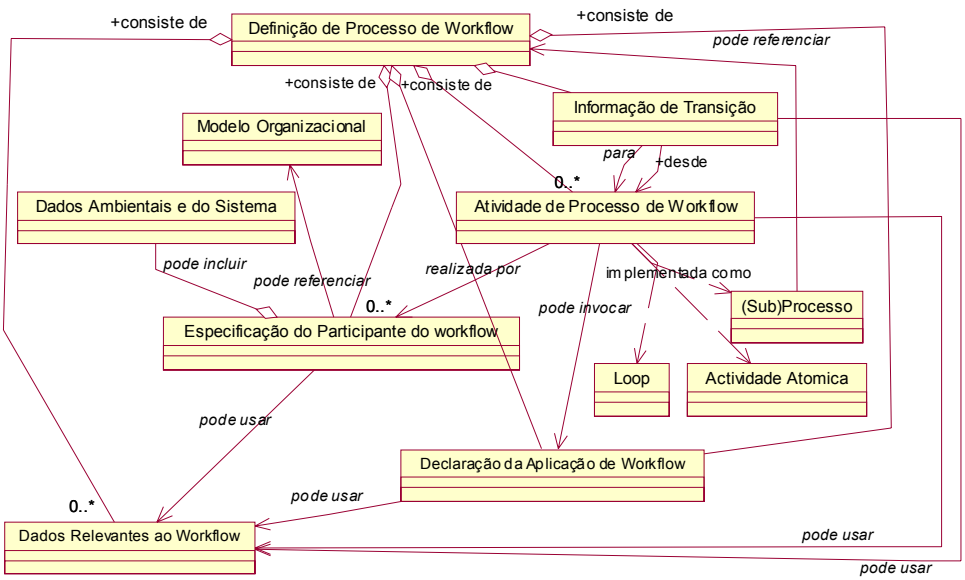


Figura 2.1 Metamodelo da definição de processo de workflow segundo WfMC

Neste modelo a definição de processo de workflow (Workflow Process Definition) contém uma ou mais atividades de processo de workflow (Workflow Process Activity), as quais estão associadas com a Declaração da Aplicação de Workflow (Workflow Application Declaration) e a Especificação do Participante do Workflow (Workflow Participant Specification) (ver definições na Tabela 2.1). As Transições (Transitions) são arcos orientados conectando atividades que levam regras do negócio para determinar o fluxo de controle no processo. O modelo provê um formalismo simples para especificar as regras do negócio em termos dos Dados Relevantes ao Workflow (Workflow Relevant Data). O metamodelo de definição de processo de workflow também inclui um modelo de recursos simples, o chamado Modelo Organizacional (Organisational Model) que contém as entidades: papel, unidade organizacional, pessoa e recurso do sistema. O metamodelo define um conjunto de propriedades para todos seus elementos. As entidades do modelo não podem ser estendidas, embora possam ser agregados atributos definidos pelo usuário às entidades do modelo, como mecanismo de extensão.

A Linguagem de Definição de Processos de Workflow da WfMC (The WfMC Workflow Process Definition Language) [2.1] define uma representação textual do metamodelo de definição de processos de workflow que corresponde à especificação da interface 1, ilustrada na Figura 1.1 do capítulo 1. A linguagem foi estabelecida para intercambiar os modelos de processo de workflow, através de um procedimento tipo 'batch' (import/export do modelo do processo). Em Maio do 2001 foi publicado pela WfMC um mapeamento da Linguagem de Definição de Processos de Workflow a XML[16] [2.6] que inclui um DTD[17] para o intercâmbio das definições de processo.

[16] XML (Extensible Markup Language) um subconjunto de SGML (Standard Generalized Markup Language) que permite a definição por parte do usuário de 'tags' específicos. Prove suporte para a representação de dados em termos de pares de atributos/valores usando os 'tags' definidos pelo usuário. [http://www.w3.org/XML]

A seguinte Tabela 2.1 descreve as entidades do metamodelo de definição de processos de workflow da WfMC.

Tabela 2.1 Descrição das entidades do Metamodelo de workflow da WfMC

Definição de Processo de Workflow (Workflow Process Definition): esta entidade descreve o próprio processo e provê outra informação opcional associada com a administração (data de criação, autor, etc.) da definição do processo ou a ser usada durante a execução do processo (parâmetros de iniciação, pessoa a ser notificada, etc). Esta entidade é um recipiente do próprio processo e provê informação que está relacionada como todas as outras entidades no processo.
Atividade de Processo de Workflow (Workflow Process Activity): uma definição de processo consiste de uma ou mais atividades cada uma representando uma unidade lógica de trabalho dentro da definição do processo. Uma atividade representa trabalho que será realizado pela combinação de recursos (especificados mediante uma Especificação do Participante do Workflow (Workflow Participant Specification)) e/ou de aplicações computadorizadas (especificadas mediante uma Declaração da Aplicação de Workflow (Workflow Application Declaration)). Estes recursos humanos e/ou máquinas são necessários para suportar a execução do processo. O uso de Dados Relevantes ao Workflow (Workflow Relevant Data) por parte da Atividade também pode ser especificado. Distinguem-se três tipos de Atividades: • Atividade Atômica (Atomic Activity): é o caso normal e corresponde à menor unidade de trabalho que é especificada dentro do processo. • Sub-fluxo (Sub-flow): é uma atividade recipiente para execução de uma definição de processo especificada separadamente e que pode ser executada localmente, dentro do mesmo serviço de workflow ou em outro serviço de workflow remoto. • Loop: é uma atividade controladora para execução repetida de um conjunto de atividades dentro da mesma definição de processo.
Informação de Transição (Transition Information): as atividades relacionam-se umas com as outras via condições de controle de fluxo (informação de transição) . Cada transição tem três propriedades: Atividade-Origem, Atividade-Destino, e a condição que deve ser satisfeita para que a transição seja realizada . Uma transição é uma simples alocação de rota. É um ponto durante a execução de uma instância do processo no qual uma atividade é completada e a linha (o thread) de controle passa para outra atividade, a qual se inicia. Uma transição pode ser governada por uma ou mais condições de transição.
Especificação do Participante do Workflow (Workflow Participant Especification): esta entidade provê a descrição dos recursos que podem agir como realizadores das várias atividades de uma definição de processo. Os recursos particulares que podem ser associados com a realização de uma atividade específica são definidos como atributos da atividade - mediante uma designação do participante , a qual liga a atividade com o conjunto de recur-

[17] DTD, Document Type Definition, prove uma definição formal de um documento, define os nomes que podem ser usados como elementos do documento, onde podem se encontrar e como eles estão relacionados dentro do arquivo de XML

sos que podem ser alocados a ela. Esse trabalho geralmente manifesta-se na forma de um ou vários itens de trabalho que são alocados ao participante do workflow via uma lista de trabalho. O metamodelo descreve quatro tipos de recursos que podem ser definidos dentro da Especificação do Participante do Workflow: Unidade Organizacional, Pessoa/humano, Papel/função e Recurso do Sistema/máquina.

Declaração da Aplicação de Workflow (Workflow Application Declaration): esta entidade provê a descrição das aplicações ou interfaces que podem ser invocadas pelo servidor de workflow para suportar ou automatizar o processamento associado com cada atividade. A Declaração da Aplicação de Workflow é especificada como um atributo(s) da atividade – designação da aplicação. A Declaração da Aplicação de workflow reflete a interface entre a máquina de workflow e a aplicação ou interface incluindo os parâmetros a serem passados.

Dados Relevantes ao Workflow (Workflow Relevant Data): esta entidade define os dados que são criados e usados durante a execução de uma instância de processo. Estes dados estão disponíveis para as atividades e para as aplicações que são executadas durante o workflow e podem ser usados para passar informação persistente ou resultados intermédios entre as atividades e/ou ser usados na avaliação das expressões condicionais nas transições ou na designação dos participantes. Os Dados Relevantes ao workflow são de um tipo particular, a linguagem de workflow define vários tipos básicos de dados. Estes dados são também usados pelo Sistema de Gerenciamento do Workflow para determinar as transições de estado nas instâncias de workflow.

Dados Ambientais e do Sistema (System & Environmental Data): estes são dados que são mantidos pelo Sistema de Gerenciamento do Workflow ou pelo ambiente do sistema local, mas que podem ser acessados pelas atividades do workflow ou usados pelo sistema de gerenciamento na avaliação de expressões condicionais da mesma maneira que os Dados Relevantes ao Workflow. Como tal, podem ser considerados como uma extensão dos Dados Relevantes ao Workflow

Modelo Organizacional (Organizational Model): em cenários mais sofisticados a Especificação do Participante do Workflow pode se referir a um modelo organizacional externo à Definição de Processo de Workflow, que permita a avaliação de expressões mais complexas que incluam referências a funções do negócio e a entidades organizacionais. O modelo organizacional representa as entidades organizacionais, seus atributos e relacionamentos.

2.1.1 Comentários

A total ausência, no modelo de definição de processo workflow da WfMC, de conceitos relacionados com eventos, limita o uso do modelo para especificar as definições de processo de workflow usando só, como mecanismo de controle do fluxo, a Informação de Transição do tipo: Atividade de Origem, Atividade Destino e a condição que deve ser satisfeita

para que a transição seja feita. Fica fora a possibilidade de especificar o controle do fluxo baseado em eventos.

A linguagem da WfMC se enfoca basicamente à sintaxe dela mais que as descrições da semântica do processo. A linguagem não define construtores de roteamento que permitam expressar a semântica de controle que envolva um mistura de escolha e sincronização de diferentes transições. Por exemplo, a falta de um mecanismo conceitual para agrupar várias transições ou um grupo de Dados Relevantes do Workflow (Workflow Relevant Data) não permite declarar condições para especificar o fato de que a realização de 'n' entre 'm' (com n <= m) transições, executando em paralelo, resultem na passagem para uma outra atividade comum, como seguinte passo dentro do workflow:

O metamodelo de definição de processos da WfMC não é um metamodelo orientado a objetos e como tal não é compatível semanticamente com os conceitos do MetaMetaModelo de referência MOF na arquitetura de metamodelos de quatro níveis da OMG apresentado na seção 1.2.1 do capítulo 1.

2.2 O METAMODELO DE WORKFLOW DA NORTEL

De acordo com o metamodelo de workflow da Nortel [2.2], um processo de workflow é visto como uma coleção de *tarefas* . Uma *tarefa* representa uma unidade de trabalho dentro de um processo de workflow. A estrutura do processo de workflow é expressa por meio de interdependências entre as tarefas que o constituem. Uma dependência pode ser uma dependência de notificação (indicando que uma tarefa só pode começar depois de outra ter terminado, dependência temporal) ou uma dependência de fluxo de dados (indicando que uma tarefa precisa das saídas de outra para poder começar). Uma tarefa é modelada como tendo *conjuntos de entrada* (input sets) e *conjuntos de saída* (output sets). Um *conjunto de entrada* especifica as *entradas* (inputs) de que a tarefa necessita (os recursos de entrada).

A execução de uma tarefa é disparada quando todo o *conjunto de entradas* estiver disponível. Um conjunto de entradas está disponível quando todas as entradas que o constituem forem satisfeitas (tenham sido obtidas, semântica AND). Basta que um conjunto de entrada (se a tarefa tiver vários conjuntos de entrada) seja satisfeito para que a execução da tarefa seja disparada (semântica OR). As *entradas* (inputs) são modeladas como objetos de dados. As *tarefas* manipulam referências aos *objetos de entrada/saída*.

Na Figura 2.2(a) é apresentado o modelo de tarefas seguindo a notação da Nortel e na Figura 2.2(b) o modelo computacional da mesma tarefa usando um diagrama de atividades da UML. Na Figura 2.2 a tarefa ti é representada com dois *conjuntos de entrada* is_1, e is_2 (InputSet_1 e InputSet_2) e dois conjunto de saída os_1 e os_2 (OutputSet_1 e OutputSet_2). O primeiro dos *conjuntos de entrada* tem dois *objetos de entrada* io_1 e io_2 (InputObject_1 e InputObject_2) e o segundo *conjunto de entrada* três *objetos de entrada* io_1, io_2 e io_3 (InputObject_1, InputObject_2 e InputObject_3). O *conjunto de entrada* is_1 (InputSet_1) será satisfeito quando obtiver os seus dois *objetos de entrada* io_1 e io_2 (se-

mântica AND) e o *conjunto de entrada* is_2, se seus três *objetos de entrada* io_1, io_2 e io_3 estiverem disponíveis. Uma dada entrada pode ser alimentada por uma ou mais fontes. Por exemplo, o *objeto de entrada* io_1 (InputObject_1), que faz parte do *conjunto de entrada* is_1 (InputSet_1), pode provir de duas fontes diferentes. Esta é a forma de introduzir , numa tarefa, fontes redundantes de dados, tendo a tarefa o controle na seleção da entrada.

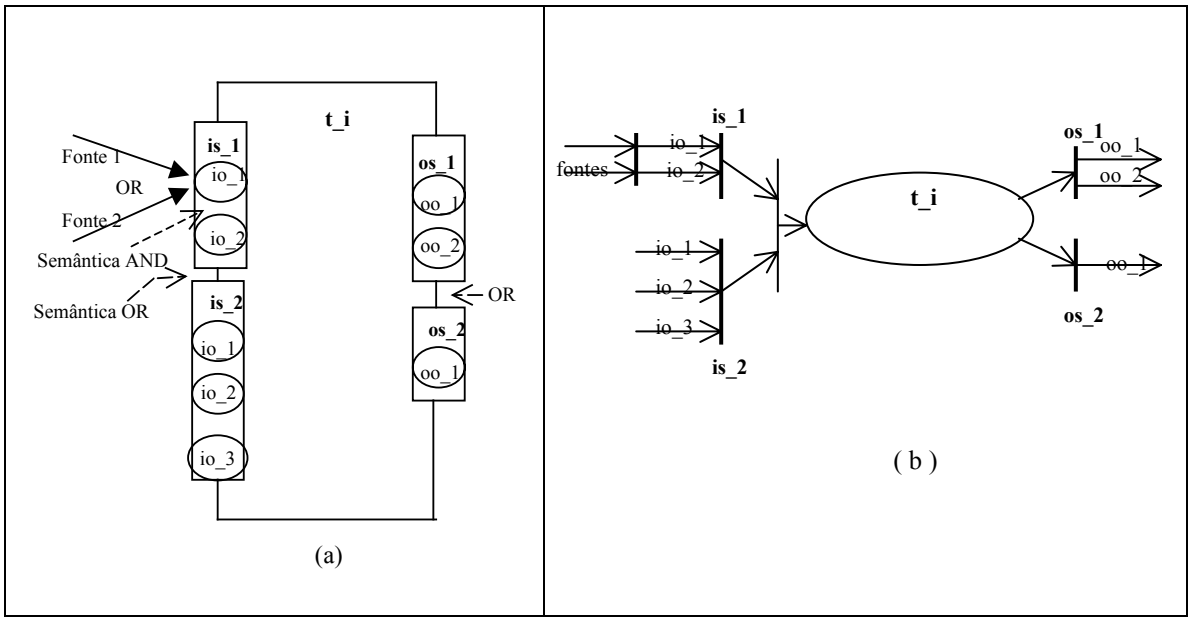


Figura 2.2. Modelo computacional de tarefa da Nortel: a) segundo a notação original da proposta b) usando um diagrama de atividades da UML.

Uma tarefa pode terminar produzindo só um *conjunto de saída* , entre vários possíveis. A tarefa da figura tem dois conjuntos de saída os_1 (OutputSet_1) e os_2 (OutputSet_2), cada um produzindo diferentes *objetos de saída.* O primeiro dos *conjuntos de saída* possui dois *objetos de saída* oo_1 e oo_2 e o segundo *conjunto de saída* os_2 (OutputSet_2) só um *objeto de saída oo_1* (OutputObject_1). Só um dos dois *conjuntos de saída,* ao ficar satisfeito, indicará o término da tarefa t_i (é a forma de representar diferentes possibilidades de terminação de uma tarefa).

Na Tabela 2.2, é apresentada uma síntese com os metaconceitos do modelo de workflow da Nortel. A definição do metamodelo de workflow da Nortel é apresentada por meio de um conjunto de metatipos e de metaassociações. Cada um dos metatipos e meta-associações é seguido por uma descrição de sua função. Para facilitar a referência com outros documentos, a descrição de cada meta-entidade citada está acompanhada pela respectiva transcrição em Inglês, tal como aparece na fonte original. Para facilitar a diferenciação do contexto de proveniência, os elementos do modelo da Nortel serão apresentados com letras em *itálico sublinhadas.*

Tabela 2.2. Metaconceitos do modelo de workflow da Nortel

Tarefa (Task): O metatipo tarefa modela as tarefas de um processo de workflow sem especificar se ela é uma *TarefaSimples*, uma *TarefaComposta*, uma *TarefaGênesis* ou uma *TarefaAdaptadora*. Uma *tarefa* é uma unidade de atividades, dentro de uma aplicação baseada em workflow. Contem um ou mais *ConjuntosDeEntrada* / *ConjuntosDeSaída*.
TarefaSimples (SimpleTask): O metatipo *TarefaSimples* deriva do metatipo *Tarefa* e modela tarefas simples (uma unidade de trabalho) dentro de um processo de workflow.
TarefaComposta (CompoundTask): O metatipo *TarefaComposta* , derivado do metatipo *Tarefa,* modela as tarefas compostas dentro de um processo de workflow. Uma tarefa composta contém (é recipiente) do metatipo *tarefa*. O propósito de uma tarefa composta é permitir a composição de tarefas a partir de outras tarefas, permitindo uma estrutura recursiva dentro do modelo do processo de workflow.
TarefaGênesis(GenesisTask): Uma tarefa gênesis é um lugar deixado para colocar a estrutura de uma tarefa. Seu propósito é permitir a ação dinâmica, criação e controle de subtarefas, a partir de uma determinada definição de tarefa.
TarefaAdaptadora (AdapterTask): A tarefa adaptadora tem uma estrutura interna desconhecida. O propósito de uma tarefa adaptadora é a construção de vias de entrada a sistemas de workflow legados.
Evento (Event): O metatipo evento modela eventos dentro de uma aplicação de workflow, em geral sem especificar se o evento é o resultado da produção de um conjunto de saída ou se resulta da seleção de um conjunto de entrada.
ConjuntoDeEntrada (InputSet): O metatipo *ConjuntoDeEntrada* deriva do metatipo *Evento*, e modela, dentro de uma aplicação com workflow, a seleção de um conjunto de entradas.
ConjuntoDeSaida (OutputSet): O metatipo *ConjuntoDeSaida* deriva do metatipo *Evento* e modela, dentro de uma aplicação com workflow, a produção de um conjunto de saídas.
Dados (Data): Este metatipo modela os dados em geral (referências a objetos), dentro de uma aplicação com workflow, sem especificar se os dados são entradas ou saídas
Entrada (Input): Este metatipo deriva do metatipo *Dados*, e modela as entradas de uma tarefa dentro de uma aplicação com workflow.
Saída (Output): Este metatipo deriva do metatipo *Dados* e modela as saídas de uma tarefa dentro de uma aplicação com workflow.
DependênciaDeDados (DataDependency): Este metatipo modela as dependências entre dados numa aplicação com workflow.

DependênciaDeEventos(EventDependency) : Este metatipo modela as dependências entre *eventos* numa aplicação com workflow.
Localização (Location): Este metatipo modela a informação necessária para especificar como são criadas as entidades dentro do workflow. Derivando-se dele outros metatipos como, por exemplo, *Pessoa* ou *Organização*, permite modelar as políticas associadas com a designação dos executores.
ControlaLocalizaçãoDe (ControlLocationsOf): Esta meta-associação modela o relacionamento entre as tarefas e as localizações requeridas para especificar a localização do controle das tarefas
LocalizaçãoDe (LoationsOf): A meta-associação *LocalizaçãoDe* modela o relacionamento entre uma tarefa simples e a localização requerida para especificar a localização da implementação da tarefa.
ConteúdoDe (ContentsOf): Esta meta-associação modela o relacionamento entre *TarefasCompostas* e as *Tarefas* requeridas para especificar as tarefas contidas na tarefa composta
EspecificaçãoDe (SpecificationOf): A meta-associação *EspecificaçãoDe* modela o relacionamento entre uma *TarefaGênesis* e a *Tarefa* requerida, para especificar a definição da tarefa que é possuída pela *TarefaGênesis.*
ConjuntosDeEntradaDe (InputSetsOf): Esta meta-associação modela o relacionamento entre *Tarefas* e os *ConjuntosDeEntrada* requeridos para especificar os conjuntos de entrada possuídos pela tarefa
ConjuntosDeSaídaDe (OutputSetsOf): Esta meta-associação modela o relacionamento entre *Tarefas* e os *ConjuntosDeSaída* requeridos para especificar os conjuntos de saída possuídos pela tarefa.
FontesDeDependênciaDeEventos (EventDependencySourcesOf): Esta meta-associação modela o relacionamento entre o meta tipo *Evento* e o meta tipo *DependênciaDeEventos* e serve para especificar a fonte de uma dependência de eventos.
PoçosDeDependênciaDeEventos (EventDependencySinksOf): Esta meta-associação modela o relacionamento entre o metatipo *Evento* e o metatipo *DependênciaDeEventos* e serve para especificar o poço de uma dependência de eventos.
EntradasDe (InputsOf): A meta-associação *EntradaDe* modela o relacionamento entre o metatipo *Entradas* e o metatipo *ConjuntoDeEntradas* para especificar as entradas que são possuídas por um conjunto de entrada
SaídasDe (OutputsOf): A meta-associação *SaídaDe* modela o relacionamento entre o metatipo *Saídas* e o metatipo *ConjuntoDeSaída* para especificar as saídas que são possuídas

por um conjunto de saída.
FontesDeDependênciaDeDados (DataDependecySourcesOf): Esta meta-associação modela o relacionamento entre o metatipo *Dados* e o metatipo *DependênciaDeDados* e serve para especificar a fonte de uma dependência de dados
PoçosDeDependênciaDeDados (DataDependenciesSinksOf): Esta meta-associação modela o relacionamento entre o metatipo *Dados* e o metatipo *DependênciaDeDados* e serve para especificar o poço de uma dependência de dados.

Usando um diagrama de classes de UML, a Figura 2.3, mostra uma adaptação do metamodelo de workflow apresentado em [2.2] e descrito na Tabela 2.2.

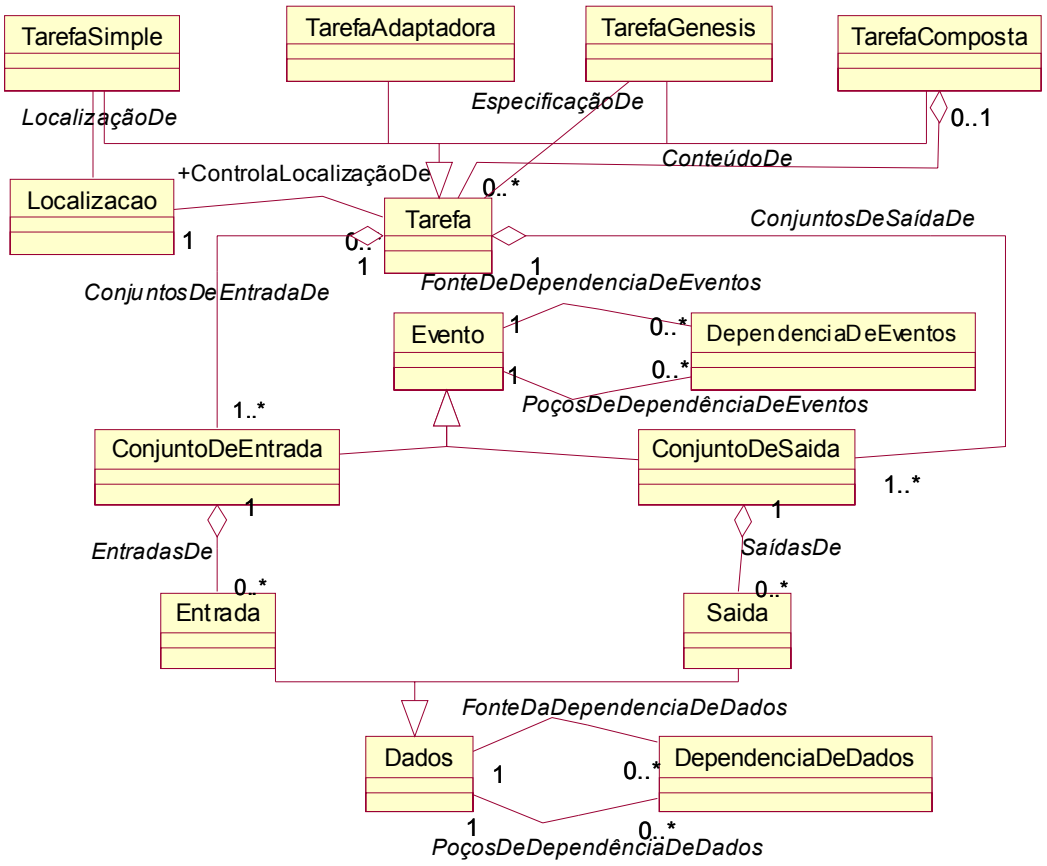


Figura 2.3 Metamodelo de workflow da Nortel

2.2.1 Comentários

O modelo de workflow da Nortel diferencia-se do modelo de workflow da WfMC por ser um modelo orientado a objetos e por incluir conceitos relacionados com eventos, embora unicamente em relação com os *ConjuntosDeEntrada* e *ConjuntosDe Saída* de uma *Tarefa*. De fato os *ConjuntosDeEntrada* e *ConjuntosDeSaída* são uma especialização da metaclasse *Evento* (ver figura 2.3) e a satisfação de qualquer conjunto de dados (o fato do conjunto ter obtido todos seus dados) gera um evento.

O fato do modelo ter metaconceitos para agrupar as *Entradas* e *Saídas* em *GruposDeEntrada* e *GruposDeSaída* possibilita a declaração de condições para especificar que a realização de 'n' entre 'm' *DependênciasDeDados,* executando-se em paralelo (com n <= m), resultem na passagem a outra *Tarefa*, como seguinte passo dentro do workflow (ver comentário da seção 2.1.1). A anterior semântica pode ser conseguida simplesmente agregando na metaclasse *Dados* um atributo extra com uma semântica associada com 'multiplicidade'. Este atributo permitiria ao *Dado* agir como uma coleção de instâncias de *Dados* e uma multiplicidade dele maior que zero especificaria que pelo menos esse número de dados deverá ser recebido para que o *Dado* seja satisfeito. Por exemplo, o atributo 'multiplicidade' igual a 'n' indicaria que a chegada de 'n' *Entradas* de 'm' *DependênciasDeDado* possíveis satisfaria o *Dado* permitindo, dessa maneira, declarar condições para especificar o fato de que a realização de 'n' entre 'm' transições executando-se em paralelo (com n <= m) resultem na passagem a uma outra atividade comum, como passo seguinte dentro do workflow.

2.3 PERFIL PARA SISTEMAS EDOC

O perfil padrão de UML para especificar Sistemas de Computação de Objetos Empresariais Distribuídos (Perfil EDOC) visa o desenvolvimento de sistemas EDOC baseados em Componentes.

Mesmo existindo no Perfil EDOC um subperfil específico para a modelagem de processos de negócio (Perfil de UML para Processos de Negócio), a modelagem completa dos processos de negócio, em particular a modelagem da Perspectiva Empresarial [18] (Enterprise View Point) em termos do Modelo de Referência de Processamento Distribuído Aberto (Reference Model for Open Distributed Processing, RM-OPD) [2.8], pode requerer o uso de outros metaconceitos que, no documento fonte [2.3], se encontram distribuídos em três

[18] A Perspectiva Empresarial descreve como o sistema suporta os processos de negócio em termos de :

- os objetivos dos processos de negócio que vão ser suportados pelo sistema,
- os passos desses processos e os relacionamentos entre os passos,
- as regras de negócio que se aplicam a esses passos,
- os papéis (roles) das entidades do sistema que suportam o processo de negócio.

Em síntese, a especificação empresarial modela a estrutura e o comportamento do sistema no contexto da organização empresarial da qual faz parte, para o qual faz uso de um modelo empresarial.

subperfis: Perfil para Processos de Negócio, Perfil para Eventos de Negócio e Perfil de Entidades.

Cada perfil é formado por um conjunto de extensões de UML, necessárias para modelar aspectos específicos de um sistema EDOC.

O perfil de Entidades descreve o conjunto de extensões usadas para modelar Objetos-Entidade, que representam conceitos no domínio da aplicação.

O perfil de Eventos descreve o conjunto de extensões necessárias para modelar sistemas controlados por eventos. Este perfil pode ser usado só ou em combinação com outros elementos que fazem parte do perfil EDOC.

O perfil para Processos de Negócio descreve o conjunto de extensões necessárias para modelar o comportamento do sistema no contexto do negócio que o sistema está suportando. Este perfil pode ser usado só ou em combinação com os outros perfis para Eventos e para Entidades. O perfil para Processos de Negócio é uma especialização de outro perfil dos sistemas EDOC, o chamado Perfil para uma Arquitetura de Colaboração de Componentes (Component Collaboration Architecture, CCA Profile). Este último perfil detalha a forma como os conceitos de classes, colaborações e grafo de atividades em UML podem ser usados para modelar a estrutura e comportamento dos componentes de um sistema. O fato anterior faz com que o perfil de Processos de Negócio herde toda a complexidade do perfil para uma Arquitetura de Colaboração de Componentes, do qual é um sub tipo.

Os processos (modelados usando o Perfil para Processos de Negócio) operam sobre os Objetos- Entidade (modelados usando o Perfil de Entidades): o fluxo do processo determina que certas operações aconteçam no modelo do domínio da aplicação (Entidades), como resultado das entradas provenientes de outros sistemas, ou de participantes humanos ou da ocorrência de eventos (modelados usando o Perfil de Eventos). Dado que os eventos podem ser publicados ou recebidos por processos ou por Entidades, o perfil de Eventos define a integração de um sistema usando eventos que controlam o processamento.

2.3.1 O Perfil para Processos de Negócio

A Tabela 2.3, mostra uma síntese com a definição dos principais conceitos de modelagem que fazem parte do perfil de UML para processos de negócio. Estes conceitos permitem a descrição dos processos de negócio em termos de:

i) uma composição de atividades,
ii) critérios para selecionar as entidades que executam essas atividades e
iii) comunicação e coordenação entre elas.

Os conceitos apresentados na Tabela 2.3 correspondem aos elementos do metamodelo que implicitamente suportam o perfil de UML para processos de negócio. Para facilitar a referência a outros documentos, a descrição de cada meta-entidade citada é acompanhada pela respectiva transcrição em Inglês, tal como aparece na fonte original.

Para facilitar a diferenciação com os conceitos que fazem parte dos metamodelos de workflow que foram introduzidos nas sessões 2.1 e 2.2, os conceitos extraídos do perfil EDOC serão escritos em **negrito**, sublinhados os que provêm do modelo da WfMC e em *itálico sublinhados* os que provêm do modelo da Nortel.

Tabela 2.3 Síntese dos conceitos do Perfil para Processo de Negócio.

ProcessoDeNegócio (BusinessProcess): Representa a especificação do processo de negócio como um todo. O **ProcessoDeNegócio** é um **ComponenteDeProcesso (Process-Component)** .Um **ComponenteDeProcesso** define uma unidade de comportamento que é configurável e pode ser usada em outras **Composições**[19] (**CCA Composition**). Através da **ComponenteDeProcesso** o **ProcessoDeNegócio** expõe **portas (ports)** que definem interfaces, operações e eventos a serem usados nas **Composições**. O **ProcessoDeNegócio** é o ponto de entrada da modelagem de processo na **Arquitetura de Colaboração de Componentes**[20] **(Component Collaboration Architecture-CCA)**. Um processo de negócio é definido por uma **TarefaComposta (CompoundTask)**. Em outras palavras, um **ProcessoDeNegócio** é visto externamente como um **ComponenteDeProcesso** e internamente como uma **TarefaComposta**.
TarefaComposta (CompoundTask): Uma **TarefaComposta** define a forma de coordenar um conjunto de **Atividades** relacionadas, que combinadas realizam uma atividade de maior escala, no contexto de um **ProcessoDeNegócio**. É também um recipiente das **Atividades**, dos **FluxosDeDados (DataFlows)** entre essas **Atividades** e dos **PapeisNoProcesso (ProcessRoles)** que modelam os enlaces com os objetos que são utilizados por essas **Atividades**.
Atividade (Activity): Representa a execução de uma parte do **ProcessoDeNegócio**. A Atividade modela uma ação que pode ser descrita na forma de outra **TarefaComposta** ou diretamente realizada por objetos interligados a instâncias da meta-entidade **PapelNoProcesso (ProcessRole).** A Atividade pode estar associada de três formas diferentes com **PapelNoProcesso (ProcessRole)**: • associação tipo, **realizadaPor (performedBy)** : representando a execução de algu-

[19] Composições: Na arquitetura de Colaboração de Componentes vários ComponentesDeProcesso podem ser usados para especificar outros ComponentesDeProcesso mais complexos, processo chamado de composição. Uma Composição pode ser representada usando os Diagramas de Colaboração de UML ou a notação própria do CCA.

[20] Arquitetura de Colaboração de Componentes ('Component Collaboration Architecture'): Esta arquitetura é composta pelo conjunto dos diferentes sub-perfis que fazem parte do Perfil EDOC e detalha como os conceitos de UML de Classe, Colaboração, e Grafo de Atividades podem ser usados para modelar a estrutura e comportamento dos componentes que fazem parte de um sistema (especificação de componentes para um processo colaborativo)

[21] Um UsoDePorta de Entrada pode ser entendido como um correlacionador de vários valores de entrada provenientes de fluxos diferentes e seria análogo a um conjunto de lugares para os parâmetros de um método os quais necessitam de um valor como argumento antes de que o método possa ser executado com algum significado verdadeiro.

ma funcionalidade de um objeto ligado a uma instância de **PapelNoProcesso**;

- associação tipo, **usaArtefato (usesArtifact);**
- associação tipo**, responsávelPor (responsibleFor)**.

A **Atividade** pode alimentar com dados as entidades usadas na execução, através do **UsoDePortas (PortUsages),** representando diferentes formas de inicialização (um **UsoDePorta** age como um correlacionador[21]). Essas entidades contêm um ou mais **ConectoresDePortasDeProcesso (ProcessPortConnectors)** representando **FluxoDeDados (DataFlows).** Cada Atividade pode ter **Pre CondiçõesDeAtividade (ActivityPreCondition(s))** e **PósCondiçõesDeAtividade (ActivityPostCondition(s))** que adicionalmente restringem o começo e término de **Atividade**.

EntidadeDeProcessoDeNegócio (BusinessProcessEntity): é um processo de negócio que também é uma entidade, com identidade própria. É usada para modelar processos de longa duração que possam requerer administração ou interação durante seu tempo de vida.

PapelNoProcesso (ProcessRole): define um lugar para colocar as entidades de um processo que realizam uma **Atividade** ou que são usadas na realização de uma **Atividade** (componentes usados pela **Atividade** para fazer o seu trabalho). Define um lugar para definir um comportamento. O proprietário de um **PapelNoProcesso** é uma **TarefaComposta** e o comportamento de **PapelNoProcesso** é parte do comportamento das **Atividades** com as quais está associado. Contém **RegrasDeSeleção**, que são expressões para buscar os objetos a serem interligados no momento de execução. O **PapelNoProcesso** está relacionado com as **Atividades** via uma associação do tipo: **realizadaPor**, e/ou **usaArtefato**, e/ou **responsávelPor.**

FluxoDeDados (DataFlow): representa um relacionamento de causa num processo de negócio. O **FluxoDeDados** também propaga valores de dados entre **ConectoresDePortasDeProcesso (ProcessPortConnectors)** que estejam relacionados. A semântica de execução é a seguinte: uma instância de **FluxoDeDados** é criada quando a instância da **TarefaComposta** que a contem é criada. A habilitação da fonte do **FluxoDeDados** causa a habilitação do **FluxoDeDados**, que então propaga os valores desde o **ConectorDePortaDeProcesso** fonte até ao **ConectorDePortaDeProcesso** destino.

ConectorDePortaDeProcesso (ProcessPortConnector): representa o uso de uma **PortaDeFluxoDeProcesso (ProcessFlowPort)** no contexto de uma **TarefaComposta**. Um **ConectorDePortaDeProcesso** pode ser fonte ou destino de um **FluxoDeDados.**

PortaDeFluxoDeProcesso (ProcessFlowPort): É a forma mais simples de **Porta** (as Portas definem os pontos de interação entre as **ComponentesDeProcesso**) a qual pode produzir ou consumir só um tipo de dados e representa os dados usados nas entradas e saídas de uma **TarefaComposta**. Uma instância de uma **PortaDeFluxoDeProcesso (ProcessFlowPort)** é representada, numa **Atividade,** por um **ConectorDePortaDeProcesso** (representa seu uso)**.** Uma **PortaDeFluxoDeProcesso** tem um atributo (multiplicidade) para indicar o menor e o maior número de valores que devem ser recebidos ou enviados pelo **ConectorDePortaDeProcesso** que instancia esta **Porta**.

PortaMultiplaDeProcesso (ProcessMultiPort): representa um conjunto relacionado de portas do tipo **PortaDeFluxoDeProcesso (ProcessFlowPort)**, usadas para descrever as entradas e saídas de uma **TarefaComposta**. A **PortaMultiplaDeProcesso** age como uma forma de correlação de **FluxoDeDados (DataFlow).** A semântica de execução associada é a seguinte: Uma instância de **PortaMultiplaDeProcesso (ProcessMultiPort)** só poderá ser ativada quando todas as instâncias de **PortaDeFluxoDeProcesso (ProcessFlowPort)** que contem forem ativadas (semântica AND). Uma **PortaMúltiplaDeProcesso** pode ser assíncrona ou síncrona.
GrupoDeEntrada (InputGroup): É uma especialização de **PortaMúltiplaDeProcesso (ProcessMultiPort)** que representa o recipiente de um número determinado de **PortasDeFluxoDeProcesso** as quais são as entradas de uma **TarefaComposta** e age como um correlacionador de **FluxoDeDados.** A semântica de execução é a seguinte: O **GrupoDeEntrada** deve ser ativado antes de que a **Atividade** possa entrar no estado "–Running".
GrupoDeSaída (OutputGroup): É uma especialização de **PortaMúltiplaDeProcesso (ProcessMultiPort)** que representa o recipiente de um número determinado de **PortasDeFluxoDeProcesso.** Representa as possíveis saídas de uma **TarefaComposta**, e age como um correlacionador dos dados associados com essa saída.
GrupoDeExcepção (ExceptionGroup) : representa o resultado de uma **TarefaComposta** que falhou em completar sua função.
Executor (Performer): O **PapelNoProcesso** tipo **Executor** é usado especificamente para identificar uma entidade que possa executar a **Atividade** com a qual está associado (realiza o trabalho da **Atividade**)
Artefato (Artifact): O **PapelNoProcesso** tipo **Artefato** é usado especificamente para identificar uma entidade usada como recurso por uma **Atividade** (é um recurso usado pelo **Executor**)
ParteResponsável (ResponsibleParty): O **PapelNoProcesso** tipo **ParteResponsável** é usado especificamente para identificar uma entidade que tem responsabilidade pela **Atividade** com a qual está associado.

A Figura 2.4 mostra, num diagrama de classes simplificado (sem incluir os atributos de cada classe), os principais conceitos que fazem parte do metamodelo de processos de negócio

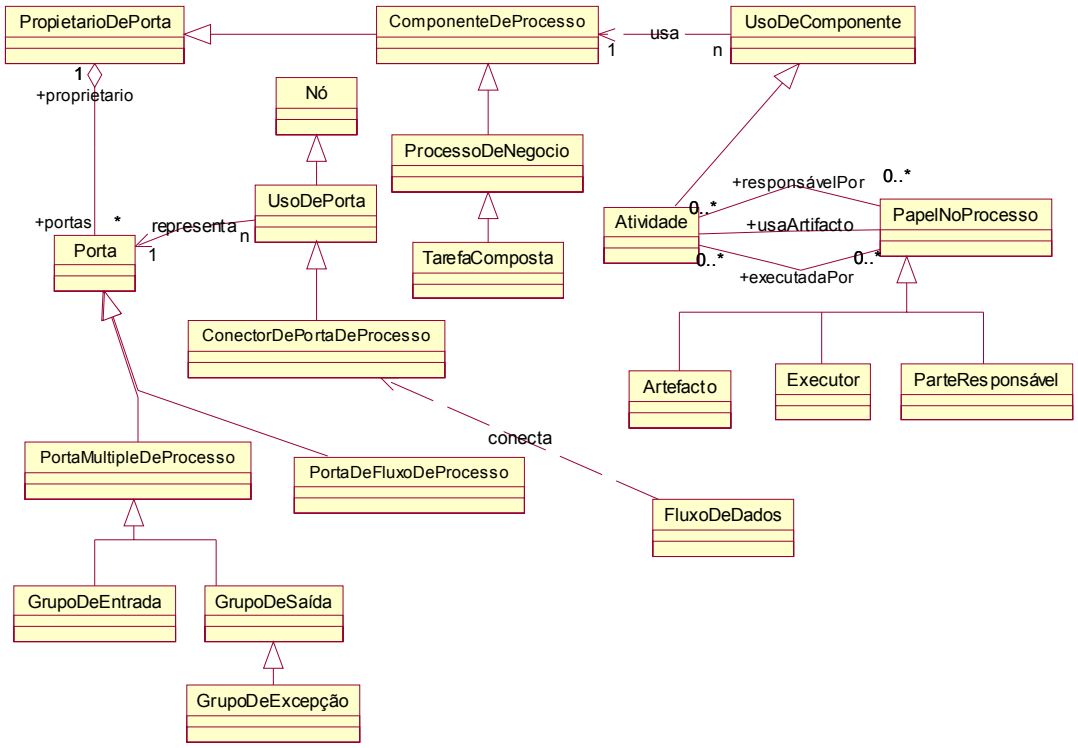


Figura 2.4 **Diagrama de classes mostrando os principais conceitos do metamodelo de Processos de Negócio segundo EDOC. (adaptado de [2.3])**

2.3.1.1 Síntese sobre o modelo para processos de negócio

Os principais elementos de modelagem usados na descrição de um processo de negócio são: **ProcessoDeNegócio**, **TarefaComposta** e **Atividades**. O **ProcessoDeNegócio** representa a especificação completa do processo de negócio. As **Atividades** representam os diferentes trabalhos requeridos para completar um processo. As **TarefasCompostas** são recipientes para um conjunto de **Atividades** e para os **FluxoDeDados** que definem as dependências temporais e de dados entre elas. Os **ConectoresDePortaDeProcesso** representam formalmente as entradas e saídas de uma **TarefaComposta**. Os **FluxosDeDados** permitem a conexão dos **ConectoresDePortaDeProcesso** de uma **TarefaComposta**. O **FluxoDeDados** tipicamente vai desde a entrada do **ConectorDePortaDeProcesso** de uma **TarefaComposta** à entrada do **ConectorDePortaDeProcesso** de uma **Atividade** contida pela **TarefaComposta**, e logo, desde o **ConectorDePortaDeProcesso** de saída da **Atividade,** ao **ConectorDePortaDeProcesso** de entrada de outra **Atividade** ou ao **ConectorDePortaDeProcesso** de saída da **TarefaComposta**.

Os **GruposDeEntrada** e os **GruposDeSaida** são usados para agregar as **PortasDeFluxoDeProcesso**, e indicam que cada uma das **PortaDeFluxoDeProcesso** pertencentes a um Grupo deve receber valores do **FluxoDeDados** antes de que algum dos valores possa ser transmitido pela **TarefaComposta** que os contem. Os **GruposDeEntrada** e de **Saída** podem ser considerados como um mecanismo de correlação.

Adicionalmente uma **Atividade** pode especificar: i) **Executores (PapelNoProcesso)** da Atividade, ii) **Artefatos (PapelNoProcesso)** que selecionem entidades requeridas como recursos e iii) **PartesResponsáveis (PapelNoProcesso)** que selecionem pessoas ou outros papeis que sejam responsáveis pela **Atividade**. Cada Atividade pode ter **PreCondiçõesDe-Atividade e PósCondiçõesDeAtividade** que adicionalmente restringem o momento de início da **Atividade** e o como ela é completada.

Os **PapeisNoProcesso** têm três classes de relacionamentos com as **Atividades**. Uma **A-tividade** pode ser **executadaPor** um **PapelNoProcesso** ou também é possível que uma **Atividade** tenha uma associação tipo **usaArtifacto** com o **PapelNoProcesso**, ou tipo **res-ponsávelPor** no caso em que um **PapelNoProcesso** seja responsável por uma **Atividade**. O mesmo **PapelNoProcesso** pode ter várias associações com diferentes **Atividades**, por e-xemplo ser o **Executor** para uma **Atividade** enquanto pode ser um **Artefato** para outra, ou ser ao mesmo tempo **ParteResponsável** e o **Executor** de uma **Atividade**. Em tempo de execução o **PapelNoProcesso** representa uma variável de estado da **TarefaComposta** que posui esse **PapelNoProcesso** e uma instância concreta que satisfaz os requerimentos esta-belecidos nos atributos do **PapelNoProcesso**. A Figura 2.5 ilustra na notação de EDOC o uso destes conceitos num diagrama de uma componente de software composta por outras três componentes de software.

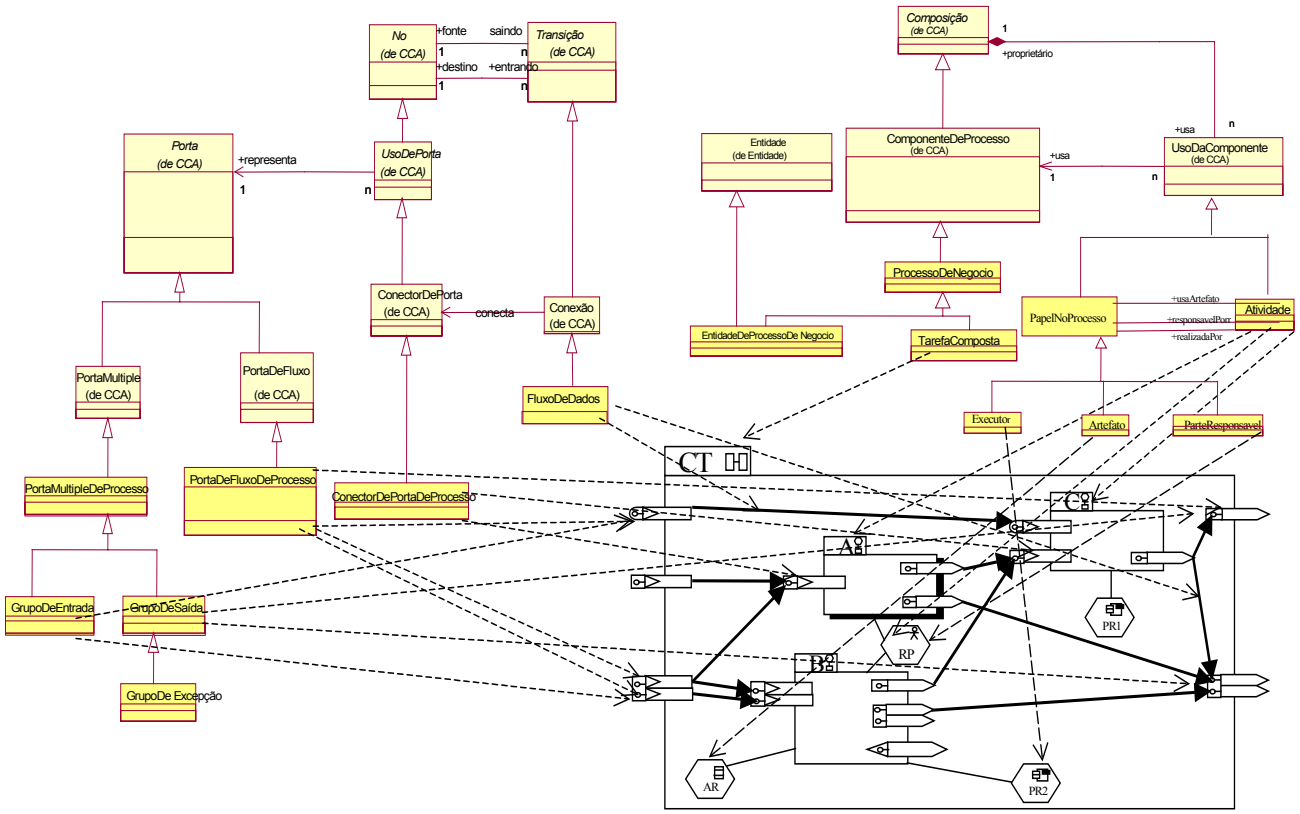


Figura 2.5 **A estrutura de um processo como um ComponenteDeProceso usando o perfil EDOC.**

2.3.2 Perfil de UML para Eventos

O perfil de Eventos descreve o conjunto de extensões ao UML que podem ser usadas para modelar sistemas controlados por eventos. Essas extensões podem ser usadas independentemente ou em combinação com outros elementos de modelagem que fazem parte dos outros sub-perfis de EDOC (como por exemplo, do perfil de Processos de Negócio e do perfil de Entidades).

Um evento em EDOC representa uma mudança de estado num sistema e que é de interesse externamente ao componente no qual acontece. Um evento pode ser definido como:

(i) uma mudança de estado que provoca que uma condição de interesse seja verdadeira, ou
(ii) pode estar associado a uma transição de um estado para outro estado particular

Quando um evento acontece é gerada uma **NotíciaDeEvento (EventNotice)** (a habilidade de gerar notas de eventos pode ser projetada dentro de um componente). O conteúdo da **NotíciaDeEvento (EventNotice)** provê informação apropriada acerca do evento. As Notícias de Eventos são publicadas ,–são enviadas-, à infra-estrutura de comunicação de eventos para serem recebidas pelos subscritores (podem existir vários subscritores ou nenhum). O **Publicador (Publisher)** de uma **NotíciaDeEvento (EventNotice)** não tem conhecimento sobre os subscritores e os subscritores não têm conhecimento sobre as fontes específicas das notícias de eventos às quais eles se subscrevem. Existem portas de publicação e portas de subscrição que provêem as interfaces para estabelecer esse relacionamento de publicação e subscrição entre componentes. As portas de publicação e subscrição podem ser agregadas às Entidades. Essas portas podem ser definidas para operar em modo síncrono ou assíncrono.

A Tabela 2.4 mostra uma síntese com a definição dos principais conceitos de modelagem que fazem parte do perfil de eventos.

Tabela 2.4 Síntese dos conceitos do Perfil de Eventos

EventoDeNegócio (BusinesssEvent): é qualquer evento de interesse ao negócio e que acontece dentro da empresa.
NotíciaDeEvento (EventNotice): é qualquer estrutura de dados que é disparada por um **EventoDeNegócio**. É comunicada como **FluxoDeDados (DataFlow)** desde os **Publicadores (Publisher)** aos **Subscritores (Subscriber)** baseados nas **Subscrições (Subscriptions).**
Publicador (Publisher): é um componente que oferece uma lista de **Publicações (Publications)** e produz ou publica **NotíciasDePublicaçãoSubscrição (PubSubNotice).** Uma **Publicação (Publication)** é o compromisso de enviar **NotíciasDePublicaçãoSubscrição (PubSubNotice). Publicação** herda de **PortaDeFluxo (FlowPort)** ou seja é uma porta de um componente**.** E uma **NotíciaDePublicaçãoSubscrição (PubSubNotice)** é uma estrutura de dados na qual as instâncias de **NotíciasDePublicaçãoSubscrição (PubSubNotice)** são

publicadas.
NotíciaDePublicaçãoSubscrição (PubSubNotice): é uma estrutura de dados na qual as instâncias de **NotíciasDePublicaçãoSubscrição (PubSubNotice)** são publicadas. As **NotíciaDePublicaçãoSubscrição (PubSubNotice)** são anunciadas por uma **Publicação (Publication)** e subscritas por um **Subscritor (Subscriber).** As instâncias de **NotíciaDePublicaçãoSubscrição (PubSubNotice)** são comunicadas como **FluxoDeDados (DataFlows),** desde os **Publicadores** aos **Subscritores,** em conformidade com as **Subscrições**.
Subscritor (Subscriber): é uma entidade que faz **Subscrições (Subscription)** e recebe (consome), de acordo com elas, **NotíciasDePublicaçãoSubscrição**. Uma **Subscrição (Subscription)** é uma declaração da capacidade ou interesse em receber uma **NotíciaDePublicaçãoSubscrição (PubSubNotice). Subscrição** herda de **PortaDeFluxo (FlowPort)** ou seja é uma porta de um componente.
RegrasDeNotificação (NotificationRules): são as regras que governam a execução de um **ProcessoBaseadoEmEventos (EventBasedProcess)** ou de partes dele. A regra está associada com uma notificação e determina o que vai acontecer dentro de um **ProcessoBaseadoEmEventos** quando a **NotíciaDeEventos** é recebida (pelo processo que faz a subscrição). Uma **RegraDeNotificação** pode ser opcionalmente protegida por uma **CondiçãoDeEvento.**
NotíciaDeEvento (EventNotice): é qualquer **NotíciaDePublicaçãoSubscrição (PubSubNotice)** que é disparada por um **EventoDeNegócio (BusinessEvent).**
CondiçãoDeEvento (EventCondition): identifica a subscrição e especifica o subconjunto (da instância da **NotíciasDePublicaçãoSubscrição)** que poderia ser recebido pelo **Subscritor** para satisfazer a condição dada pela **CondiçãoDeEvento**.
ProcessoBaseadoEmEventos (EventBasedProcess): É um **Subscritor** e tem **RegrasDeNotificação** (NotificationRules) associadas com sua **Subscrição.** É também um **Publicador (Publisher)** e publica **EventosDeProcesso (ProcessEvents).**
EventosDeProcesso (ProcessEvents): é qualquer evento de negócio que represente uma mudança de estado dentro de um processo. Os **EventosDeProcesso** descrevem o ciclo de vida dum **ProcessoBaseadoEmEventos** o que é igual à entrada ou saída dos nós de uma coreografia.[22]

2.3.2.1 Síntese do modelo de negócios baseado em eventos

[22] Coreografia (conceito definido no perfil para uma Arquitetura de Colaboração de Componentes, Component Collaboration Architecture, CCA Profile) é um conjunto de nós (estados e usos de portas) e a conexão entre elas. Uma coreografia representa o momento em que são enviadas as mensagens (o quando) entre os componentes enquanto um protocolo ilustra os mensagens que um componente envia ou recebe quando colabora com outro componete.

Um **ProcessoBaseadoEmEventos (EventBasedProcess)** é uma especialização de uma coreografia. Uma coreografia (conceito definido no perfil para uma Arquitetura de Colaboração de Componentes, Component Collaboration Architecture, CCA Profile) é um conjunto de nós (estados e usos de portas) e a conexão entre elas. Um **ProcessoBaseadoEmEventos (EventBasedProcess)** gera **EventosDeProcesso (ProcessEvents)** quando entra ou sai com sucesso (ou sem ele) de seus nós. Um P**rocessoBaseadoEmEventos** como **Publicador (Publisher)** que é, publicará **NotíciaDeEvento (EventNotice)** para cada um de seus **EventosDeProcesso**, e como **Subscritor (Subscriber)** que é fará **Subscrições** da **NotíciaDePublicaçãoSubscrição (PubSubNotice)** de outros processos ou entidades. As **RegrasDeNotificação (NotificationRules)** são as regras que governam a execução das partes do **ProcessoBaseadoEmEventos**. Uma **RegraDeNotificação (NotificationRule)**, associada com uma **Subscrição**, determina o que deveria acontecer dentro de **ProcessoBaseadoEmEventos** que faz a **Subscrição** quando a **NotíciaDeEventos** é enviada. Uma **RegraDeNotificação** pode ser opcionalmente protegida por uma ou mais **CondiçãoDeEvento (EventCondition).** Uma **CondiçãoDeEvento (EventCondition)** é uma dependência relacionada com o recebimento de outras **NotíciasDeEvento** relacionadas mas governadas por outras **Subscrições**.

Na Figura 2.6 é apresentada uma adaptação do diagrama de classes do metamodelo de Eventos apresentado em [2.3].

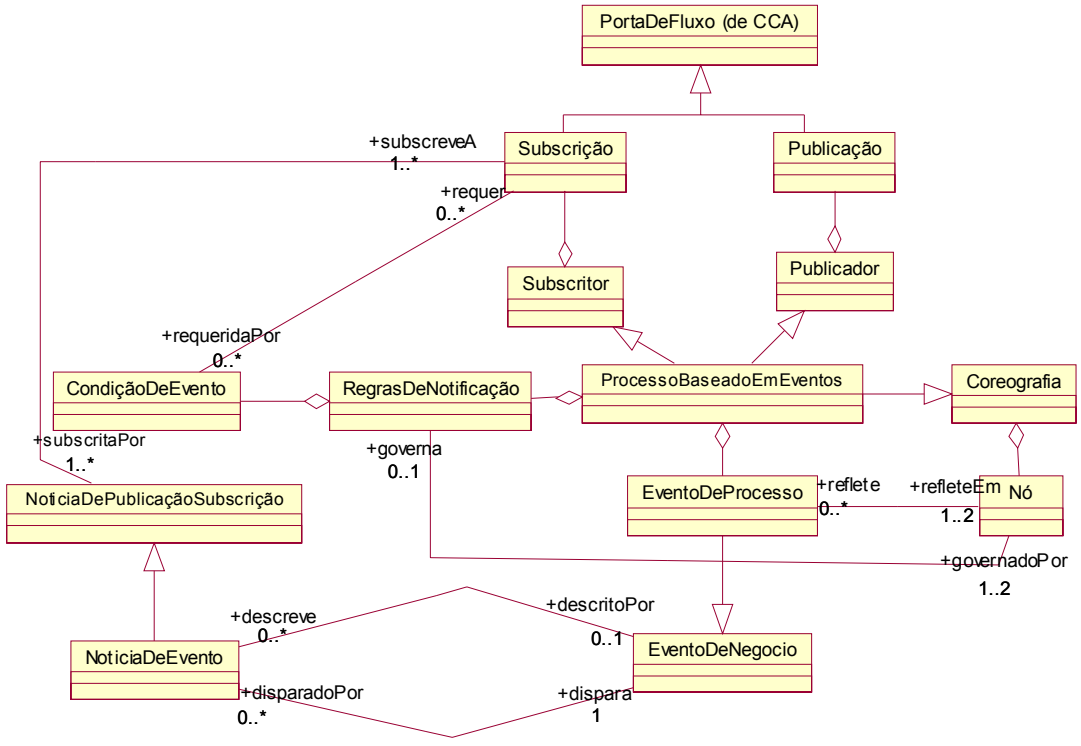


Figura 2.6 Diagrama de classes do metamodelo de eventos

2.3.3 Comentários gerais sobre deficiências e vantagens do perfil EDOC

Na introdução a esta tese (capítulo 1) foi mencionado que as especificações produzidas pelos analistas de negócio geralmente não tinham o nível de detalhe necessário para permitir aos engenheiros de software a implementação de soluções adequadas aos requerimentos do sistema a ser suportado. A riqueza conceitual do atual perfil EDOC supera claramente as necessidades conceituais para a descrição de definições de processos de workflow. A afirmação anterior ficará fundamentada quando forem apresentadas, na seção 2.4, as comparações entre os metamodelos de workflow e o metamodelo de processos de negócio do Perfil EDOC. As comparações permitirão estabelecer os conceitos ausentes nos metamodelos de workflow avaliados, embora potencialmente necessários na definição dos processos de workflow.

Apesar da riqueza conceitual do perfil EDOC, têm aparecido na literatura algumas críticas ao Perfil EDOC [2.3], recentemente aprovado como 'draft'. Em particular Bock e Ramackeres [2.9], [2.10] manifestaram algumas críticas que podem basicamente ser classificadas em três tipos.

a) Cumprimento de Requerimentos: alguns dos requerimentos obrigatórios apresentados na RFP [2.11] não foram considerados.
b) Alcance da proposta: alguns tópicos cobertos na proposta não fazem parte dos requerimentos da RFP.
c) Uso do UML: uso não apropriado da semântica do UML.

Na seção 2.3.3.1 é apresentada uma síntese com os argumentos anotados na última categoria c) para justificar a conveniência de derivar o metamodelo unificado a ser proposto no capítulo 3 (seções 3.2 e 3.3) a partir das metaclasses do Grafo de Atividades de UML 1.4 [2.12] como foi efetivamente derivado na apresentação feita seção 3.4. do capítulo 3.

A derivação do metamodelo unificado a partir do Grafo de Atividades de UML gera a necessidade de redefinir (com respeito à semântica de EDOC) a semântica de alguns dos metaconceitos do metamodelo unificado, para não violar os relacionamentos entre as metaclasses da UML 1.4. Essa redefinição gera algumas incoerências entre os dois metamodelos, que serão posteriormente enunciadas na seção 3.4.1 'Incoerências entre o metamodelo de EDOC e o metamodelo unificado'

2.3.3.1 Uso não apropriado da semântica do UML no modelo EDOC

- O propósito dos perfis é formalizar como a UML deve ser usada em vários domínios específicos de aplicação e em diferentes plataformas. O propósito dos perfis não é definir inteiramente, em paralelo à UML, uma linguagem de modelagem independente da plataforma e do domínio de aplicação, *o que está sendo feito* no perfil para processos de negócio EDOC .
- O metamodelo de componentes da proposta viola o principio básico para estereotipar UML. Embora cada metaclasse do metamodelo de componentes da proposta mapeie-se a um estereótipo de um dos Elementos de UML (descrevendo certamente a hierarquia de especialização), a hierarquia do estereótipo contradiz a hierarquia dos Elementos do

UML que estão sendo estereotipados. UML permite a especialização de um estereótipo (que já é a especialização de uma classe base) a partir de outro estereótipo (também a especialização de outra classe base) só quando as classes base de esses estereótipos possuem uma especialização similar. Ver Figura 2.7 adaptada de [2.9].

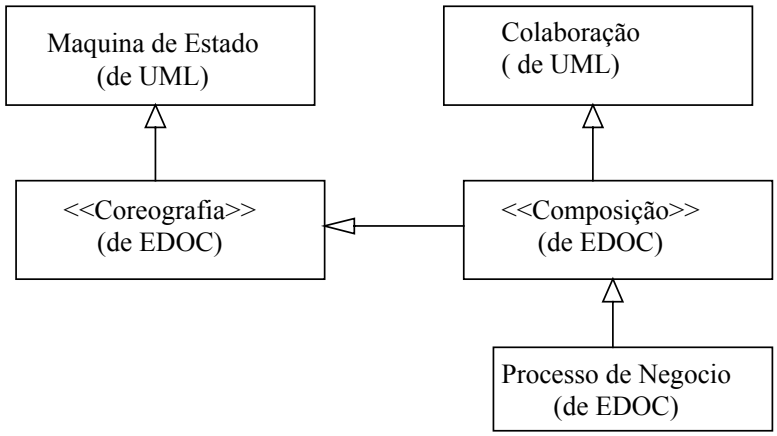


Figura 2.7 Uso não apropriado do UML em EDOC segundo argumentos extraídos de [2.9]

O perfil EDOC que modela os Processos de Negócio como uma especialização do estereótipo <<Composição>> de EDOC viola os princípios para estereotipar em UML (em UML Colaboração da UML não é uma especialização de Máquina de Estado da UML, mas no perfil EDOC <<Composição>> de EDOC especializa <<Coreografia>> de EDOC. (ver Figura 2.7).

Além do anteriormente dito, <<Composição>>, como um estereótipo que é do metaconceito Colaboração de UML, limita a generalidade da modelagem (modelo abstrato do processo que não implique uma forma de implementação) dos processos de negócio, pois o único comportamento dado aos Papeis_de_uma_Associação (AssociationRole de UML) dentro de uma Colaboração do UML é de que as Mensagens possam ser transmitidas através deles, o que é um enfoque de implementação. Em outras palavras, o perfil EDOC derivando Processos de Negócio da metaclasse Colaboração do UML (através do estereótipo <<Composição>>) especificamente usa a transmissão de mensagens entre os componentes do processo, o que é uma forma de implementação de um processo, mas não um modelo abstrato dele.

Assim, o modelo de Colaboração de UML (como uma forma de modelar o processo baseado na responsabilidade de partes jogando papéis) possibilita a modelagem do controle de um processo, seja na forma de um controle distribuído ou na forma de um controle centralizado, mas não pode modelar uma abstração de ambos tipos de controle, já que o modelo de colaboração especifica, em detalhe, a transmissão das mensagens mostrando se elas estão saindo de um controlador global ou estão sendo passadas entre controladores distribuídos (peers).

Resumo da crítica

Em resumo, embora a proposta esteja tecnicamente expressa como um perfil de UML, ela expande o modelo central de UML (UML core model) o que a transforma realmente numa extensão de UML não muito apropriada. Assim, em princípio qualquer coisa poderia ser modelada como um perfil, já que sempre é possível estender a metaclasse **Element** do UML (a metaclasse mais abstrata em UML, raiz da qual se derivam os demais conceitos da linguagem UML). Mas a intenção dos perfis é assegurar o máximo reuso da linguagem, evitando fragmentar o padrão. Mais grave é o fato de que alguns dos estereótipos da proposta violem o princípio para estereotipar em UML. Além disso, a generalidade de representação de um processo de negócio, baseado num metaconceito de Colaboração do UML, fica seriamente comprometida.

2.4 COMPARAÇÕES ENTRE OS META MODELOS

Nesta seção são comparados entre si os metaconceitos que fazem parte do perfil EDOC com os metamodelos de workflow da WfMC e da Nortel, introduzidos nas seções 2.1 e 2.2. A comparação está inspirada nos princípios propostos por Muhler [2.5] para comparar sistemas de workflow com base nos seus metamodelos. Estes princípios são introduzidos na seção 2.4.1.

2.4.1 Identificação e resolução de conflitos

Mulher [2.5] sugere determinar todos os possíveis conflitos entre os modelos a serem comparados, antes de iniciar a avaliação deles, e identifica três tipos de conflitos: (i) conflitos de nome (ii) conflitos de tipo e (iii) conflitos estruturais.

(i) Conflitos de nome: este tipo de conflito acontece quando entre os nomes, por convenção usados nos modelos a serem comparados, aparecem sinônimos[23] ou homônimos[24]. Este é o tipo mais comum de conflito encontrado na comparação de meta modelos. A identificação dos sinônimos requer uma análise dos diferentes termos com igual significado e as mesmas associações(conceito de 'semelhança'). A presença de homônimos dificulta as comparações, dado que o significado do termo está associado com um contexto particular e não pode ser determinado em geral (conceito de 'dessemelhança').

(ii) Conflito de tipo: esta classe de conflito acontece quando o mesmo fato é representado semanticamente de maneira correta em dois modelos usando conceitos diferentes.

(iii) Conflito estrutural: este tipo de conflito surge se dois metamodelos a serem integrados representam o mesmo fato usando semânticas diferentes. Este tipo de conflito,

[23] Um sinônimo é encontrado se *a mesma entidade* é identificada *com diferentes expressões* em diferentes metamodelos

[24] Um homônimo existe se *entidades diferentes* de dois ou mais metamodelos *têm o mesmo nome*.

que identifica uma violação à 'correitude' semântica, surge geralmente quando pessoas diferentes estão envolvidas na modelagem de um mesmo fato.

Ao comparar metamodelos os seguintes elementos podem prover informação útil:

(i) os diferentes tipos de entidades encontrados nos metamodelos,
(ii) os diferentes tipos de relacionamentos entre os diferentes tipos de entidades,
(iii) a cardinalidade dos relacionamentos e
(iv) os atributos das entidades.

2.4.2 Comparações

Ao fazer as comparações entre os metamodelos ter-se-á presente a informação que descreve as entidades, relacionamentos, cardinalidade e atributos de cada um deles e que se encontra consignada nas Tabelas 2.1, 2.2, 2.3, e 2.4 e Figuras 2.1, 2.3, 2.4 e 2.5.

Lembrando a convenção adotada para facilitar a diferenciação do contexto do qual provem cada um dos conceitos a serem comparados, os conceitos extraídos do perfil EDOC serão escritos em **negritos**, sublinhados os que provêm do modelo da WfMC e em *itálico sublinhados* os que provêm do modelo da Nortel.

O metaconceito **ProcessoDeNegócio (BusinessProcess)** especializa (ver Figura 2.4) o metaconceito **ComponenteDeProcesso(ProcessComponent)**. Uma **ComponenteDeProcesso** (a qual está definida no subperfil de EDOC que permite especificar uma Arquitetura de Colaboração de Componentes[25]- Component Collaboration Architecture-CCA) representa uma unidade de processamento ativa (que faz algo). Ela pode definir um conjunto de **Portas (Ports)** através das quais interage com outras **ComponentesDeProcesso** (**ProcessComponent)** e definir também um conjunto de propriedades (por meio de atributos) que podem ser usadas para configurá-la quando for usada.

Dado que **ProcessoDeNegócio (BusinessProcess)** é uma especialização de um **ComponenteDeProcesso (ProcessComponent)** e **TarefaComposta (CompoundTask)**, por sua vez, é uma especialização de **ProcessoDeNegócio** (ver Figura 2.4), as características particulares de um **ComponenteDeProcesso** são herdadas tanto pela meta-entidade **ProcessoDeNegócio** como pela meta-entidade **TarefaComposta**.

Sendo realmente estritos conceitualmente estas características herdadas (as que provêm da meta-entidade **ComponenteDeProcesso**) criam um conflito de um novo tipo (talvez poderíamos chamá-lo de conflito de herança) e diferente dos considerados nas seção 2.4.1, impedindo que os metaconceitos do Perfil EDOC sejam estrita e diretamente mapeáveis a

[25] Arquitetura de Colaboração de Componentes ('Component Collaboration Architecture'): Esta arquitetura é composta pelo conjunto dos diferentes sub-perfis que fazem parte do Perfil EDOC e detalha como os conceitos de UML de Classe, Colaboração, e Grafo de Atividades podem ser usados para modelar a estrutura e comportamento dos componentes que fazem parte de um sistema (especificação de componentes para um processo colaborativo)

algum dos metaconceitos que fazem parte dos metamodelos de workflow da WFMC e da Nortel.

Dado que a definição de um processo de workflow não precisa da complexidade envolvida na especificação de um sistema EDOC, na perspectiva do desenho dele e usando um modelo de componentes, poderíamos deixar de considerar, de maneira provisória e com o objetivo de possibilitar a comparação, alguns aspectos semânticos específicos, envolvidos com um **ComponenteDeProcesso (ProcessComponent)** e fazer o mapeamento mostrado na Tabela 2.5.

A Tabela 2.5 mostra uma matriz de comparação entre os conceitos que fazem parte do perfil EDOC, do metamodelo de workflow da WfMC e do metamodelo de workflow da Nortel. As filas na matriz identificam os conceitos sendo comparados e as colunas o contexto de procedência.

Tabela 2.5 Matriz de comparação dos metamodelos EDOC, WfMC e Nortel

Do Perfil De UML Para Processos De Negócio E Eventos- (Perfil EDOC)	Do MetaModelo Para Definição de Processos de Workflow da WfMC	*Do MetaModelo De Workflow da Nortel*
ProcessoDeNegócio (BusinessProcess)	Não definido	*Não definido*
TarefaComposta (CompoundTask)	Definição de Processo de Workflow (Workflow Process Definition)	*TarefaComposta (CompoundTask)*
Atividade (Activity)	Atividade de Processo de Workflow (Workflow Process Activity)	*TarefaSimples(SimpleTask)*
FluxoDeDados (DataFlow)	Informação de Transição (Transition Information)	*DependênciaDeDados (DataDependency) (entre instâncias)*
PortasDeFluxoDeProcesso (ProcessFlowPort) (corresponde à declaração das portas)	Não definido	*Dados (Data) Entrada(Input) / Saída (Output) (declarados como classe abstrata)*
ConectorDePortaDeProcesso (ProcessPortConnector) (corresponde ao uso das portas)	Dados Relevantes ao Workflow (Workflow Relevant Data)	*Dados (Data) Entrada(Input) / Saída (Output) (Dados como instância - uso)*
ConectorDePortaDeProcesso-	Dados Relevantes ao	*Entrada(Input) / Saída*

DeEntrada (input ProcessPortConnector) ConectorDePortaDeProcesso-DeSaída (output ProcessPortConnector)	Workflow (Workflow Relevant Data)	*(Output) (como instância da classe abstrata Dados (Data))*
PortasMultiplesDeProcesso(ProcessMultiPort)	Não definido	*Não definido como meta-classe abstrata de ConjuntoDeEntrada (InputSet) e ConjuntoDe Saída (OutputSet)*
GrupoDeEntrada(InputGroup)	Não definido	*ConjuntoDeEntrada (InputSet)*
GrupoDeSaída (OutputGroup)	Não definido	*ConjuntoDeSaída (OutputSet)*
GrupoDeExceções (ExceptionGroup)	Não definido	*Não definido*
PapelNoProcesso (ProcessRole)	1)Especificação do Participante do workflow (Workflow Participant Specification) 2) Declaração da Aplicação de Workflow (Workflow Application Declaration)	*Localização*
Executor (Performer)	Especificação do Participante do workflow (Workflow Participant Specification) tipo Pessoa/Humano	*Não definido*
Artefato (Artifact)	1)Dados Relevantes ao Workflow (Workflow Relevant Data) 2) Especificação do Participante do workflow tipo: Sistema/maquina	*Não definido*
ParteResponsável (ResponsibleParty)	Especificação do Participante do workflow (Workflow Participant Specification)/ tipo: 1) Unidade Organizacional 2) Papel/Função	*Não definido*
EventoDeNegócio (BusinesssEvent)	Não definido	*Não definido*
NotíciaDeEvento (EventNotice)	Não definido	*Não definido*
Publicador (Publisher)	Não definido	*Não definido*
NotíciaDePublicaçãoSubscrição	Não definido	*Não definido*

(PubSubNotice)		
Subscritor (Subscriber)	Não definido	*Não definido*
RegrasDeNotificação (NotificationRules)	Não definido	*Não definido*
NotíciaDeEvento (EventNotice)	Não definido	*Não definido*
CondiçãoDeEvento (EventCondition)	Não definido	*Não definido*
ProcessoBaseadoEmEventos (EventBasedProcess)	Não definido	*Não definido*
AdministradorDeDadosBaseadoEmEventos (EventBasedDataManager).	Não definido	*ConjuntoDeEntrada (InputSet)* e *ConjuntoDe Saída (OutputSet)*
EventoDeDados (DataEvent)	Não definido	*Evento(Event)*
EventosDeProcesso (ProcessEvents)	Não definido	*Não definido*

Numa primeira comparação entre os metaconceitos dos modelos de workflow e do Perfil EDOC, ressalta o fato deste último ter um número maior de diferentes tipos de entidades, o que é de fato um indicador da sua riqueza conceitual . O perfil EDOC é bastante rico semanticamente e provê elementos diferentes que permitem especificar, por separado, a descrição das características de um **ComponenteDeProcesso (ProcessComponent)** do uso ou instanciação dela por parte de outro componente. Uma definição de um processo de workflow que esta enfocada à representação da lógica do processo num modelo independente da plataforma de execução e não na forma como ela é executada não precisa dessa diferenciação.

2.4.2.1 Alguns Comentários sobre o Mapeamento

O metaconceito **ProcessosDeNegócio** (**BusinessProcess),** definido como a especificação do processo de negócio como um todo e como uma unidade de comportamento configurável, que expõe portas com interfaces, operações e eventos para poder ser usada em outras composições, não é diretamente mapeável em nenhum dos conceitos disponíveis nos metamodelos de workflow da WFMC e da Nortel. Esta situação configura o conflito que chamamos antes de 'conflito de herança' e que só poderíamos resolver deixando de considerar alguns aspectos semânticos específicos envolvidos com a semântica de um **ComponenteDeProcesso (ProcessComponent)**.

O conceito de **Atividade** (**Activity**), como representando uma ação (um uso de um componente) mapeia-se (tendo em consideração a solução dada ao conflito de herança) no metaconceito *tarefa* da Nortel, que modela em geral ações dentro de uma aplicação, baseada em workflow. O metamodelo de Processos de Negócio de EDOC não distingue explicitamente entre diferentes tipos de **Atividades** (já que os componentes podem ser configurados antes do seu uso) como é feito no modelo da Nortel, que distingue três tipos de tarefas: *Ta-*

refaSimples (SimpleTask), *Tarefa Gênesis (GenesisTask)* e *TarefaAdaptadora (Adapter-Task).* Esta diferenciação dos tipos de tarefa facilita, na definição de um processo de workflow, a especificação de diferentes detalhes a serem levados em conta no momento da automatização e controle do processo por um sistema de gerenciamento de workflow.

No metamodelo da Nortel os conceitos que representam uma ação dentro de uma aplicação baseada em workflow e a entidade que executa a ação estão precariamente diferenciados, no sentido de se encontrarem no metamodelo da Nortel englobados dentro do meta-conceito *tarefa (Task)*. Só existe a possibilidade de diferenciar as entidades executoras através da representação dos diferentes tipos de *tarefa* e a localização delas e através do metaconceito *Localização* (*Location*). No perfil EDOC a representação da ação e a execução da ação estão claramente separados. A execução é realizada 'at run time' por objetos que estão ligados com uma instância da meta-entidade **PapeisNoProcesso (ProcessRole)** com a qual uma **Atividade (Activity)** está associada. No modelo da Nortel essa semântica associada com **PapeisNoProcesso (ProcessRole)** está compreendida no seu meta elemento *Localização* (*Location*). Uma **Atividade** (**Activity**) está associada com um **Executor** via a associação do tipo **realizadaPor (performedBy).** Também pode estar associada com um ou mais **PapeisNoProcesso** via as associações **usaArtefato (usesArtifact),** para identificar uma entidade que seja necessária como recurso, pela **Atividade,** e **responsávelPor (responsibleFor)** para identificar a entidade que tem a responsabilidade dela. Isto estes três tipos de associações e os respectivos subtipos de **PapeisNoProcesso** (i) **Executor (Performer),** (ii) **Artefato (Artifact)** e (iii) **Parte Responsável (ResponsibleParty)** não tem análogos no metamodelo da Nortel. Em outras palavras o metamodelo de workflow da Nortel não tem a forma de especificar, em detalhe, diferentes tipos de recursos relacionados com a execução das *tarefas*. Nesse sentido, o modelo da Nortel está sub especificado, e o modelo de processos de negócio de EDOC orienta na necessidade de que o metamodelo unificado possa especificar estes aspectos. Nos aspectos acima considerados não existem conflitos de nome ou de tipo entre os dois modelos, simplesmente um menor número de meta-entidades num dos metamodelos. O mapeamento entre **Atividade**, Atividade de Processo de Workflow e *TarefaSimples* é semanticamente direto (corresponderia à solução do 'conflito de nome' do tipo sinônimo).

O **FluxoDeDados (DataFlow)** que representa um relacionamento de causa num processo de negócio e que também propaga valores entre **ConectoresDePortasDeProcesso (ProcessPortConnectors) -** as quais representam o uso de **PortaDeFluxoDeProcesso (ProcessFlowPort) –** relacionados, mapeia a *DependênciaDeDados (DataDependency)* metatipo da Nortel, que modela as dependências entre *Dados* dentro de uma aplicação com workflow, e parcialmente mapeia-se com a meta-entidade Informação de Transição (Transition Information) da WfMC que relaciona umas atividades com outras (Atividade-Origem, Atividade-Destino). O conceito de Informação de Transição (Transition Information) da WfMC é bastante mais rico do que o conceito de *DependênciaDeDados (DataDependency)* da Nortel e de **FluxoDeDados (DataFlow)** de EDOC, no sentido de ter algumas propriedades a mais, como as condições que devem ser satisfeitas para a transição ser feita. O metamodelo da Nortel não provê meta-entidades para especificar este tipo de condições e EDOC o provê no perfil de processos de negócio, através do atributo **preCondição (preCondition),** que faz pare da meta-entidade **Transição (Transition)**, herdada por **ProcessoDeNegó-**

cio (BusinessProcess) a partir de seu supertipo **ComponenteDeProcesso (ProcessComponent)** que, por sua vez, a herdou de **Coreografia (Choreography).**

Uma **PortaMúltiplaDeProcesso (ProcessMultiPort)** como a representação de um conjunto relacionado de portas - **PortaDeFluxoDeProcesso (ProcessFlowPort)** – é usada para descrever as entradas e saídas de uma **TarefaComposta** por meio das especializações **GrupoDeEntrada(InputGroup)** e **GrupoDeSaída (OutputGroup)** (ver Figura 2.4).

Dado que a semântica de execução dos *ConjuntoDeEntrada (InputSet)* e *ConjuntoDeSaída (OutputSet)* ser satisfeita, quando todas suas entradas estiverem disponíveis, esses conjuntos mapeiam-se nos metaconceitos **GrupoDeEntrada(InputGroup)** e **GrupoDeSaída (OutputGroup),** como especializações que são de uma **PortaMúltiplaDeProcesso(ProcessMultiPort).**

Alem da semântica anterior, quando um *ConjuntoDeEntrada/ConjuntoDeSaída* é satisfeito, gera-se um evento, o que é explicitamente representado no metamodelo da Nortel fazendo com que estes metaconceitos (*ConjuntoDeEntrada/ConjuntoDeSaída*) especializem a metaclasse *Evento* (*Event*). No metamodelo de processos de negócio EDOC, embora se tenha a mesma semântica de evento associada à meta-entidade **PortaMúltiplaDeProcesso(ProcessMultiPort),** esta última não aparece explicitamente associada com uma meta-entidade de Evento, que de fato existe, com o nome de **EventoDeNegócio (BusinessEvent),** num outro subperfil de EDOC (Perfil de UML para Eventos). Ali é definido o conceito de **EventoDeNegócio (BusinessEvent)** como qualquer evento de interesse ao negócio e que acontece dentro da empresa. Distinguem-se duas especializações dele (i) **EventoDeProcesso (ProcessEvent)** e (ii) **EventoDeDados (DataEvent)**, o primeiro para representar um evento de negócio que reflita uma mudança de estado dentro de um processo, e o segundo que reflita mudanças nos dados de um **AdministradorDeDadosBaseadoEmEventos (EventBasedDataManager)**. Este último representa um componente que tem como funcionalidade prover acesso e realizar operações sobre seus dados associados. De conformidade com a anterior semântica, os *ConjuntoDeEntrada* e *ConjuntoDeSaída* da Nortel mapeam também a semântica associada com um **AdministradorDeDadosBaseadoEmEventos** de EDOC, o qual gera **EventoDeDados (DataEvent)**. No entanto **EventoDeDados (DataEvent)** reflete a semântica da meta-entidade *Eventos (Event*) da Nortel. Aliás, a nível de execução no metamodelo da Nortel, os *ConjuntoDeEntrada* (*inputSet*) e *ConjuntoDeSaída* (*OutputSet)* pertencentes a uma *Tarefa (Task)* são controlados através da interface especificada como "*SimpleTaskController*" que oferece uma funcionalidade semelhante à semântica envolvida com a meta-entidade **AdministradorDeDadosBaseadoEmEventos (EventBasedDataManager).**

O fato anterior das entidades *ConjuntoDeEntrada/ConjuntoDeSaída* Ter em sua semântica associada a duas meta-entidades do modelo EDOC, **GrupoDeEntrada/GrupoDeSaída** e **AdministradorDeDadosBaseadoEmEventos,** é o resultado do maior número de meta-entidades no modelo EDOC que permite uma clara separação de diferentes funcionalidades (na Nortel, o agrupamento de entradas e a geração de eventos estão ligados numa mesma entidade). Neste caso particular, existe um conflito estrutural, dada a procedência independente dos dois meta-modelos, o que não se traduz num problema na configuração de um metamodelo unificado, dado que os dois modelos não vão ser agregados um com o outro.

O fato do metamodelo da Nortel restringir a semântica de eventos só às entidades *ConjuntoDeEntrada/ConjuntoDeSaída* que especializam a meta-entidade *Evento* limita, só a elas, a possibilidade de receber eventos com a semântica específica já definida para eles, o que limita a possibilidade de especificar outro tipo de eventos associados com outras entidades do modelo como, por exemplo, as *tarefas* . Para tal torna-se necessária uma adequada agregação de novos metaconceitos e seus respectivos relacionamentos no metamodelo original da Nortel.

O caso do metamodelo de definição de processos de workflow da WfMC é ainda mais criticamente restrito, nesse sentido, dada a total ausência de conceitos análogos ao agrupamento de entradas, saídas e semântica de execução associada com eventos, como está consignado nas entradas da Tabela 2.5 com a expressão 'não definido'.

Dadas as mencionadas restrições ao uso dado ao conceito de eventos nos metamodelos de workflow, os demais conceitos que fazem parte do perfil de eventos de EDOC, listados na continuação, não têm conceitos com semântica análoga a os metamodelos de workflow: **EventoDeProcesso (ProcessEvent) NotíciaDeEvento (EventNotice), NotíciaDePublicação Subscrição (PubSubNotice), Publicador (Publisher), Subscritor (Subscriber), RegrasDeNotificação (NotificationRules), NotíciaDeEvento (EventNotice), CondiçãoDeEvento (EventCondition), ProcessoBaseadoEmEventos (EventBasedProcess),** e **EventosDeProcesso (ProcessEvents)**. Estes metaconceitos serão usados para estender a potencialidade de que outros tipos de eventos, diferentes de mudanças de estado, (semântica associada com os *ConjuntosDeEntrada/Saída*), possam ser produzidos ou consumidos pelas entidades que representam os passos de um processo de workflow e/ou por suas respectivas entidades executoras.

Se, de fato, os mapeamentos apresentados constituem uma solução aos 'conflitos de nome' do tipo sinônimo (diferentes nomes associados com uma mesma semântica), os conceitos ausentes em algum ou alguns dos metamodelos considerados servirão para orientar a configuração do metamodelo unificado a ser apresentado no capítulo 3.

A próxima Tabela 2.6 apresenta uma síntese com os metaconceitos do perfil EDOC que estão ausentes nos metamodelos de workflow considerados.

Tabela 2.6 Metaconceitos do perfil EDOC não existentes nos metamodelos de workflow da WfMC e da Nortel

AUSENTES NO MODELO da WfMC	AUSENTES NO MODELO da NORTEL
ProcessoDeNegócio (BusinessProcess)	**ProcessoDeNegócio (BusinessProcess)**
PortasMúltiplasDeProcesso(ProcessMultiPort)	**PortasMúltiplasDeProcesso(ProcessMultiPort)**
GrupoDeEntrada(InputGroup)	**(existe um meta tipo análogo)**
GrupoDeSaída (OutputGroup)	**(existe um meta tipo análogo)**

GrupoDeExcepções (ExceptionGroup)	**GrupoDeExcepções (ExceptionGroup)**
(existe um meta tipo análogo)	**Executor (Performer)**
(existe um meta tipo análogo)	**Artefato (Artifact)**
(existe um meta tipo análogo)	**ParteResponsável(ResponsibleParty)**
EventoDeNegócio (BusinesssEvent)	**EventoDeNegócio (BusinesssEvent)**
NotíciaDeEvento (EventNotice)	**NotíciaDeEvento (EventNotice)**
Publicador (Publisher)	**Publicador (Publisher)**
NotíciaDePublicaçãoSubscrição (PubSubNotice)	**NotíciaDePublicaçãoSubscrição (PubSubNotice)**
Subscritor (Subscriber)	**Subscritor (Subscriber)**
RegrasDeNotificação (NotificationRules)	**RegrasDeNotificação (NotificationRules)**
NotíciaDeEvento (EventNotice)	**NotíciaDeEvento (EventNotice)**
CondiçãoDeEvento (EventCondition)	**CondiçãoDeEvento (EventCondition)**
ProcessoBaseadoEmEventos (EventBasedProcess)	**ProcessoBaseadoEmEventos (EventBasedProcess)**
AdministradorDeDadosBaseadoEmEventos (EventBasedDataManager).	**(existe um meta tipo análogo)**
EventoDeDados (DataEvent)	**(existe um meta tipo análogo)**
EventosDeProcesso (ProcessEvents)	**EventosDeProcesso (ProcessEvents)**

Conclusões do Capítulo

Este capítulo apresentou uma síntese dos metamodelos usados como referência para guiar a proposta de um metamodelo unificado para modelar processos de workflow e seu software de suporte. Os metamodelos usados como referência foram comparados entre si com o objetivo de detectar metaconceitos presentes no perfil EDOC mas ausentes nos metamodelos de workflow analisados. A comparação de esses metamodelos em si mesma é uma contribuição parcial desta tese. Os resultados de esta comparação servem de contexto para guiar a proposta de um metamodelo unificado, consistente com o perfil EDOC, a ser formalmente apresentado no Capítulo 3.

2.5 REFERÊNCIAS DO CAPÍTULO

[2.1] Interface 1 – Process Definition Interchange- Process Model, v 1.1WfMC-TC-1016-P, Workflow Management Coalition, October 29, 1999; [http://www.wfmc.org]

[2.2] Warne John, “Nortel Revised Submission to the Workflow Management RFP”, OMG Document Number: bom/98-03-01; 1998.[ftp://ftp.omg.org/pub/docs/bom/98-03-01.pdf]

[2.3] UML Profile for Enterprise Distributed Object Computing-EDOC Draft Adopted Specification OMG Document Number: ptc/2001-12-04; 2001.
[ftp://ftp.omg.org/pub/docs/ptc/01-12-04.pdf]

[2.4] Request for Proposals for a Workflow Management Facility, OMG document Number: cf/97-05-06; 1997.[ftp://ftp.omg.org/pub/docs/cf/97-05-06.pdf]

[2.5] Michael zur Muhlen. “Evaluation of Workflow Management Systems Using Meta Models”. Proceedings of the 32nd Hawaii Internacional Conference on System Sciences, 1999, (HICSS-1999)

[2.6] Workflow Process Definition Interface-XML Process Definition Language. Document Number WfMC-TC-1025. Version 0.03 (Draft), May 22, 2001

[2.7] Warne John, “Nortel Revised Submission to the Workflow Management RFP”, OMG Document Number: bom/98-03-01, 1998.[ftp://ftp.omg.org/pub/docs/bom/98-03-01.pdf]

[2.8] ISSO/IEC&ITU-T: Information Technology – Open Distributed Processing – Enterprise Viewpoint – ITU-T Recomendation X.911-ISSO/IEC 15414.

[2.9] Conrad Bock,Guus Ramackers . Partial Evaluation of the Joint EDOC Submission, August 20, 2001. OMG Document Number: ad/01-08-26; 2001
[ftp://ftp.omg.org/pub/docs/ad/01-08-26.pdf]

[2.10] Conrad Bock, Thomas Weigert, Response to Response to the Evaluation of the Joint EDOC Submission, August 27, 2001 . OMG Document Number: ad/01-08-33; 2001
[ftp://ftp.omg.org/pub/docs/ad/01-08-33.pdf]

[2.11] “UML Profile for Enterprise Distributed Object Computing”, –Request for Proposal, 1999, OMG Document Number: ad/99-03-10; 1999
[http://ftp.omg.org/pub/docs/ad/99-03-10.pdf]

[2.12] Editor, Ms. Linda Heaton. “Unified Modeling Language Specification v1.4 “; 2001 OMG Document Number: formal/01-09-67; 2001 a formal/01-09-80; 2001 .[http://ftp.omg.org/pub/docs/formal/01-09-67.pdf]

[2.13] Juergen Boldt. Workflow Management Facility Specification (WMF), v1.2, 2. OMG Document Number: formal/00-05-02; 2000.
[http://ftp.omg.org/pub/docs/formal/00-05-02.pdf]

CAPÍTULO 3

O METAMODELO UNIFICADO E A NOTAÇÃO TEXTUAL

3.1 INTRODUÇÃO

Neste capítulo serão apresentados os principais resultados que fazem parte da tese. A partir da matriz de comparação de conceitos, apresentada no capítulo anterior é proposto um metamodelo unificado (MMU) que suporta tanto a definição de processos de workflow quanto o projeto do sistema de software que dá o suporte à automação dos processos de negócio associados.

Para a execução de uma definição de processo de workflow, por um sistema de gerenciamento de workflow, supõe-se a existência de um algoritmo que faça a análise da representação gráfica da definição do processo de workflow, modelada com base no metamodelo unificado proposto, e que gere automaticamente um esquema de workflow, num formato de linguagem textual, para que possa ser reconhecido, interpretado e executado pela máquina de workflow. Neste capítulo é também apresentada uma proposta dessa representação textual.

Na seção 3.2 apresenta-se o metamodelo unificado. Na seção 3.3 é detalhada a semântica de cada um dos elementos do metamodelo unificado. A seção 3.4, de conformidade com os argumentos expostos no capítulo 2, seção 2.3.3, deriva o metamodelo unificado a partir das metaclasses do Grafo de Atividades da UML 1.4 e introduz o mapeamento de uma definição de processo de workflow em termos da UML. A seção 3.5 propõe uma representação textual correspondente à modelagem que usa o metamodelo unificado.

3.2 O METAMODELO UNIFICADO

O resultado da comparação dos metaconceitos que fazem parte do Perfil EDOC com os modelos de workflow da WfMC e da Nortel, apresentados no capítulo 2, permite estabelecer as seguintes observações gerais:

a) A semântica do metamodelo de processos de negócio do perfil de UML para sistemas EDOC é bem similar à semântica usada no modelo de workflow da Nortel.

EDOC descreve o processo de negócio através do conceito **TarefaComposta (CompoundTask)** que age como um recipiente para um conjunto de **Atividades (Activities)** que representam as ações que fazem parte do processo de negócio. Para representar a informação necessária do processo, tanto as **TarefasCompostas,** quanto as suas **Atividades** constituintes, contêm **GruposDeEntrada** e **GruposDeSaída** como mecanismo de correlação dos valores a serem transmitidos, os quais são representados por **FluxoDeDados.** As Atividades especificam os recursos necessários para sua execução através dos

conceitos **Executor, Artefato,** e **PartesResponsáveis,** todos eles especializações do metaconceito **PapelNoProcesso.**

O modelo de workflow da Nortel, descreve um processo de workflow também através do metaconceito *TarefaComposta (CompoundTask*) como recipiente do meta tipo *tarefa* que representa uma unidade de trabalho num processo de workflow. Também, para representar as *Entradas (inputs)* e *Saídas (ouputs)* de dados, tanto da *TarefaComposta,* quanto das *Tarefas*Simples, as *tarefas* contêm *ConjuntosDeEntrada (input sets)* e *ConjuntosDeSaída (output sets)* como mecanismo de correlação dos dados. A estrutura do processo de workflow é expressa por meio de *DependênciaDeDados* (indicando que uma tarefa precisa das saídas de outra para poder começar).

A diferença substancial, do ponto de vista semântico, entre o metamodelo do perfil EDOC e o metamodelo de workflow da Nortel (sem ter em consideração a riqueza semântica e complexidade que o perfil de processos de negócio EDOC herda do perfil Arquitetura de Colaboração de Componentes, quando a entidade **ProcessoDeNegócio** especializa um **ComponenteDeProcesso**) foi descrita no capítulo 2 seção 2.4.2. Vale ressaltar:

- a ausência no metamodelo da Nortel de um metaconceito para descrever a designação dos recursos que uma *tarefa* precisa para sua execução;
- o modelo da Nortel liga a semântica de eventos só aos relacionadores de dados, os *ConjuntosDeEntrada (input sets)* e *ConjuntosDeSaída (output sets)* e
- a ausência de meta-elementos no metamodelo da Nortel para representar condições sobre as dependências de dados (*DependênciaDeDados)*.

b) Os processos de negócio podem ser baseados em eventos e o perfil EDOC provê os correspondentes metaconceitos para suportar a modelagem através do sub-perfil de eventos.

A automação desses processos de negócio, baseados em eventos, exige um correspondente metamodelo de workflow que suporte os conceitos de eventos no contexto da definição de um processo de workflow. O metamodelo de workflow da WfMC não inclui os conceitos de eventos. O metamodelo de workflow da Nortel inclui um conceito de eventos bastante restrito. Fica clara a necessidade de incluir os conceitos de eventos no metamodelo unificado.

As observações enunciadas em a) e b), sugerem os elementos mínimos necessários para, a partir do metamodelo da Nortel, estabelecer um metamodelo unificado que permita modelar tanto a definição de processos de workflow quanto o sistema de software que dá o suporte à automação do processo de negócio e que seja totalmente coerente com o perfil EDOC. Em particular as seguintes tarefas precisam ser executadas:

(i) desvincular no modelo unificado o conceito de *evento* da Nortel dos metaconceitos *ConjuntoDeEntradas (input sets)* e *ConjuntoDeSaídas (output sets)*;

(ii) incluir no metamodelo unificado os conceitos de eventos de forma consistente com o perfil de eventos de EDOC (eventos de negócio) e especializando a semântica deles no contexto da definição de um processo de workflow (eventos de workflow);

(iii) incluir no metamodelo unificado conceitos para representar a designação de recursos que uma *tarefa* pode precisar para sua execução.

Tendo em consideração todas as observações anteriores, propõe-se o metamodelo unificado (MMU), ilustrado na Figura 3.1.

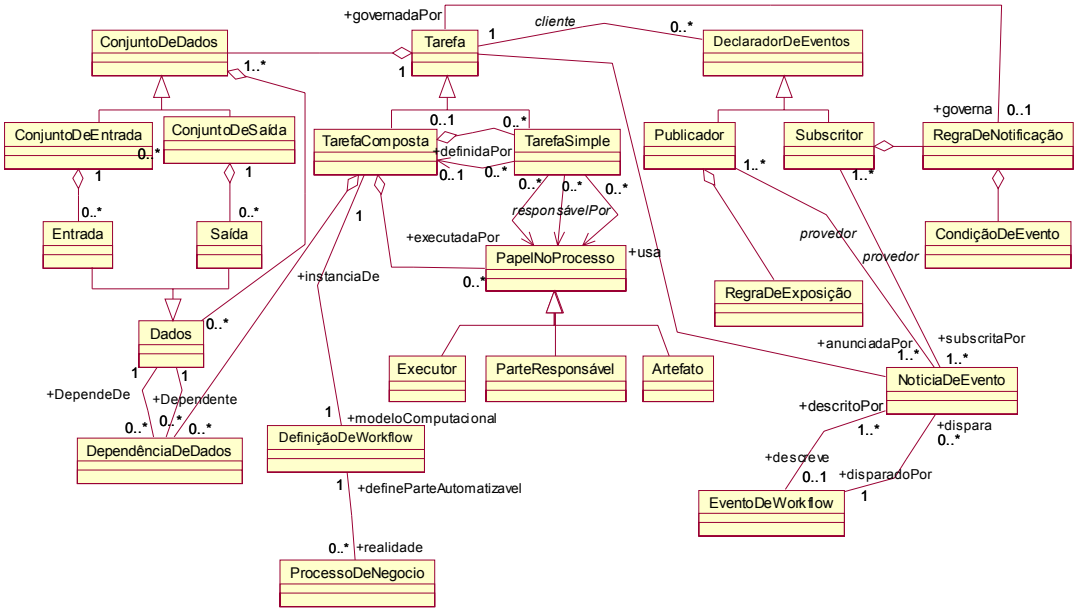


Figura 3.1 Metamodelo unificado para processos de negócio e workflow.

3.3 SEMÂNTICA DO METAMODELO UNIFICADO

Para facilitar a diferenciação, do contexto de procedência dos metaconceitos que fazem parte dos diferentes metamodelos, os conceitos provenientes do perfil de UML para EDOC foram escritos em **negrito**, sublinhados os que provêm do modelo da WfMC, em *itálico sublinhados* os do modelo da Nortel e em **negrito sublinhado** os que fazem parte do metamodelo unificado proposto.

De acordo com o metamodelo unificado proposto um processo de negócio (**ProcessoDeNegócio)** estabelece o contexto dentro do qual as atividades do negócio se realizam de maneira coordenada, para alcançar um objetivo de negócio. O processo de negócio existe na realidade e está associado com a definição de workflow (**DefiniçãoDeWorkflow**) que define a sua parte automatizável (role **defineParteAutomatizavel**). Uma instância de execução de uma definição de workflow é definida através do metaconceito de tarefa composta (**TarefaComposta)**. A **TarefaComposta** está constituída de tarefas mais simples (**TarefaSimples)** que representam as ações do processo de negócio. Tanto o processo (a **Tarefa-**

Composta) como as **TarefasSimples** (as ações) que o constituem contêm conjuntos de dados de entrada (**ConjuntoDeEntrada**) necessários para a execução, e produzem conjuntos de dados de saída (**ConjuntoDeSaída**), como resultado da execução. Igualmente elas podem declarar a potencialidade de produzir (**Publicador**) eventos de workflow (**EventoDeWorkflow**) ou de consumi-los (**Subscritor**). As tarefas simples (**TarefasSimples)** podem ser vistas como definindo unidades atômicas de execução e as tarefas compostas (**TarefasCompostas**) como sub-processos que agrupam, de maneira lógica, um conjunto de tarefas simples (**TarefasSimples)**, dentro de outro processo de maior nível hierárquico. Desta maneira, o processo de maior nível hierárquico também pode ser considerado como uma tarefa composta (**TarefaComposta)**. Uma **TarefaSimples** pode ser descrita através de uma **TarefaComposta** via a associação **DefinidaPor,** permitindo-se, mediante está associação, especificar a utilização recursiva de sub processos.

No metamodelo unificado, as tarefas simples (**TarefasSimples)** para realizar seu trabalho podem precisar de recursos especiais (**Artefatos**), ou ser diretamente executadas por outras entidades (**Executores** e/ou **ParteResponsável**) que satisfaçam as restrições explicitadas na meta-entidade **PapelNoProcesso** via as associações **executadaPor** ou **responsávelPor** respectivamente. A meta-entidade **PapelNoProcesso** permite fazer a conexão entre o processo de workflow e as entidades necessárias para realizar cada passo do processo. O proprietário de um **PapelNoProcesso** é uma **TarefaComposta** e o comportamento de **PapelNoProcesso** é parte do comportamento das **TarefasSimples** com as quais está associado.

As dependências entre as **Tarefas** (ações) as quais determinam a sequência de execução das Tarefas pode definir-se de duas formas:

(i) uma, especificada pela dependência (**DependênciaDeDados**) entre os dados produzidos (**Saídas**) por uma Tarefa e os dados de entrada (**Entradas**) que são necessários para execução de outra **Tarefa**;

(ii) a outra através de regras de comportamento que analisam eventos usando **CondiçõesDeEvento** que permitem especificar as mudanças no processo que têm que ver com a entrada ou com a saída de uma **Tarefa** . Esta declaração de condições faz parte do consumidor do evento, o chamado **Subscritor** do evento.

Uma análise da Figura 3.1 permite observar que o metamodelo unificado proposto é composto de três grandes blocos:

1) No bloco da esquerda os metaconceitos estão relacionados com a declaração da informação (dados) que uma tarefa precisa: **ConjuntoDeEntrada**, **ConjuntoDeSaída**, **ConjuntoDeDados**, **Entradas**, **Saídas, Dados**, e **Dependência DeDados.**
2) Na parte central do metamodelo encontram-se os metaconceitos a serem usados para declarar as tarefas que fazem parte do processo de workflow e as entidades necessárias para a execução de cada um dos passos. **Tarefa** como classe abstrata é especializada por **TarefaSimples** e **TarefaComposta**, as quais estão relacionadas com as entidades executoras via **PapelNoProcesso** e suas especializações **Executor, Artefato** e **ParteResponsável**.
3) No bloco direito estão representados os metaconceitos **Publicador**, **Subscritor**, **DeclaradorDeEventos**,**NotíciaDeEvento**, **EventoDeWorkflow**, **RegraDeNotificação,**

RegraDeExposição e **CondiçãoDeEvento**, os quais têm a ver com a possibilidade de se especificar eventos por parte das **Tarefas**.

A seguir descreve-se, em detalhe, o propósito e semântica de cada um dos elementos ilustrados na Figura 3.1 e que fazem parte do metamodelo unificado.

3.3.1 ConjuntoDeEntrada

Modela o conjunto de valores que uma **Tarefa** requer para fazer um trabalho. Modela os valores que a **Tarefa** precisa para entrar no estado 'executando-se'(ver máquina de estado associada com uma Tarefa em seção 3.3.8). Herda da metaclasse **ConjuntoDeDados**. Um **ConjuntoDeEntradas** é o recipiente de uma ou mais **Entradas**. Um **ConjuntoDeEntradas** está contido numa **Tarefa** e é satisfeito quando todos seus valores tenham sido recebidos (semântica AND) o que habilita a **Tarefa** a entrar no estado 'executando-se'.

3.3.2 ConjuntoDeSaída

Modela os possíveis resultados produzidos pela execução de uma **Tarefa** e é uma especialização de **ConjuntoDeDados**. Um **ConjuntoDeSaídas** é o recipiente de uma ou mais **Saídas**. Um **ConjuntoDeSaídas** está contido numa **Tarefa** e é satisfeito quando todos seus valores de saída tenham sido produzidos (semântica AND) o que indica ao mesmo tempo que a **Tarefa** entrou no estado 'parado' (ver máquina de estado associada com uma Tarefa em seção 3.3.8)

3.3.3 ConjuntoDeDados

Representa, em geral, um conjunto de dados relacionados que são usados para descrever as **Entradas** e **Saídas** de uma **Tarefa** e tem como especializações **ConjuntoDeEntradas** e **ConjuntoDeSaídas**. Um **ConjuntoDeDados** age como um correlacionador de várias **DependênciasDeDados,** é o recipiente de zero ou mais **Dados** e está contido dentro de uma **Tarefa**.

3.3.4 Entrada

Modela a informação de dados que uma **Tarefa** pode consumir. **Entrada** herda de **Dados** e está contido num **ConjuntoDeEntrada**. Uma **Entrada** fica satisfeita quando pelo menos o número de dados de entrada recebidos é igual ao número estabelecido no seu atributo 'multiplicidade' (ver **Dados** na seção 3.3.6). Uma **Entrada** que faz parte de um **ConjuntoDeEntradas** que pertença a uma **TarefaSimples,** pode ser **DependenteDe** uma ou mais **DependênciasDeDados**. A **Entrada** pode então ser alimentada por uma ou mais **DependênciasDeDados,** até ao valor de sua 'multiplicidade'. Uma **Entrada** que faz parte de um **ConjuntoDeEntradas** que pertença a uma **TarefaComposta** pode ser fonte de uma ou mais dependências tipo **Dependente**.

3.3.5 Saída

Modela a informação de dados que uma **Tarefa** pode produzir. **Saída** herda de **Dados** e está contido num **ConjuntoDeSaída**. Uma **Saída** fica satisfeita quando ao menos o número de dados de saída produzidos é igual ao número estabelecido no seu atributo 'multiplicida-

de' (ver **Dados** na seção 3.3.6). Uma **Saída** que faça parte de um **ConjuntoDeSaídas** que pertença a uma **TarefaComposta** pode ser **DependenteDe** uma ou mais **DependênciasDeDados.** Uma **Saída** que faça parte de um **ConjuntoDeSaídas** que pertença a uma **TarefaSimples** pode ser fonte de uma ou mais dependências tipo **Dependente.**

3.3.6 **Dados**

Representa, em geral, de maneira abstrata, a informação de dados de entrada ou de saída de uma **Tarefa** . **Dados** está contido num **ConjuntoDeDados.** Este elemento tem dois subtipos **Entrada** e **Saída**. Um dado seja de **Entrada** ou de **Saída** está só contido por um **ConjuntoDeDados**. Uma **Entrada** está só contida num **ConjuntoDeEntradas** e uma **Saída** num **ConjuntoDeSaída.** Um dado está disponível quando todos os dados de seu **ConjuntoDeDados** estiverem disponíveis. **Dados** tem um atributo 'multiplicidade', este atributo permite a um **Dado** agir como uma coleção de instâncias de **Dados**, uma multiplicidade maior que zero especificaria que ao menos esse número de dados deve ser recebido para o **Dado** ficar satisfeito. Por exemplo, o atributo 'multiplicidade' igual a 'n' indicaria que a chegada de 'n' **Entradas** de 'm' **DependênciasDeDado** possíveis satisfaria o **Dado**. Este atributo permite a declaração de condições para especificar que a realização de 'n' de 'm' **DependênciasDeDados** executando-se em paralelo (com n <= m) resultem na inicialização de outra **Tarefa** como passo seguinte no workflow. Os dados podem ter um tipo de dados associado.

3.3.7 **DependênciaDeDados**

Representa uma relação de causalidade no processo de workflow. A fonte da dependência (**dependeDe**) deve acontecer antes do destino da dependência (**dependente**). As **DependênciaDeDados** propagam os valores de dados entre **Dados** relacionados e estão contidas em **TarefaComposta**. Uma **TarefaComposta** pode conter zero ou mais **DependênciaDeDados**, os quais são criados no momento em que a **TarefaComposta** é instanciada. A habilitação de uma fonte de dependência (**Dados** no papel de **dependeDe**) habilita uma **DependênciaDeDados**, o que causa a propagação dos valores, desde a fonte até ao destino (**Dados** no papel de **dependente**). Esta metaclasse contem dois meta- atributos 'pré_ ativação' e 'pós_ ativação'. As condições lógicas de saída, que a tarefa fonte deve cumprir, para que o fluxo de controle seja ativado, são declaradas através do atributo de 'pós-ativação'. As condições lógicas que a tarefa destino deve cumprir para o que o fluxo de controle seja ativado são declaradas através do atributo 'pré-ativação'.

3.3.8 **Tarefa**

Uma **Tarefa** representa, em geral, uma unidade de trabalho. É recipiente direto de seus **ConjuntoDeEntradas** e **ConjuntoDeSaídas** e indiretamente de suas respectiva **Entradas** e **Saídas**. **Tarefa** tem como especializações **TarefaSimples** e **TarefaComposta** e tem associada a máquina de estado ilustrada na Figura 3.2 , coerente com a máquina de estado associada com a interface *WfExecutionObject* que faz parte da especificação de interfaces para uma Facilidade de Workflow da OMG [3.2]

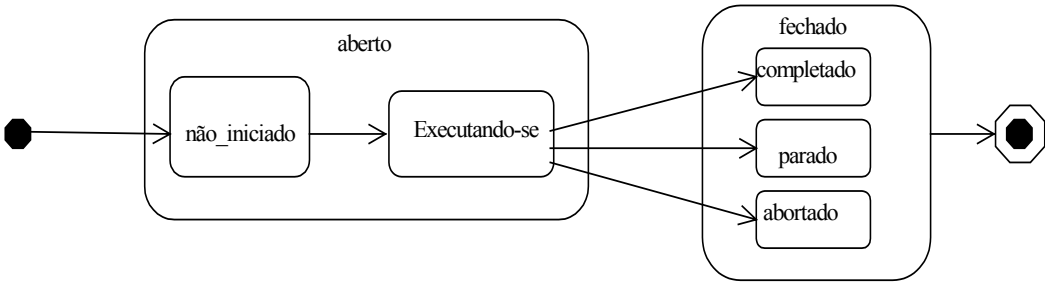


Figura 3.2 Máquina de estado associada com Tarefa

3.3.9 TarefaSimples

Representa uma ação que faz parte de uma **DefiniçãoDeWorkflow**. A ação é executada diretamente pelos objetos que especializam a entidade **PapelNoProcesso** e que estão associados com ela (via a associações **executadaPor** e **responsávelPor**) ou ela cria uma instância de uma **TarefaComposta** que a descreve mais em detalhe (via a associação **Definida-Por)** .Quando uma **TarefaSimples** é instanciada, entra no estado 'não iniciado'. Uma vez habilitado o seu **ConjuntoDeEntradas** sua associação com o **PapelNoProcesso** é resolvida, e a **TarefaSimples** entra no estado 'executando-se'. Durante este estado os valores de **Entrada** podem ser usados (consumidos). Quando os todos os valores de **Saída** de um dos seus **ConjuntoDeSaídas** forem produzidos **TarefaSimples** entra no estado 'parado' e logo depois no estado 'completado'.

3.3.10 TarefaComposta

A **TarefaComposta** representa como um todo uma **instânciaDe** uma definição de processo de workflow (**DefiniçãoDeWorkflow)**. **TarefaComposta** define a coordenação do conjunto de **TarefasSimples** relacionadas que em combinação representam o modelo computacional (**modeloComputacional)** da definição do processo de workflow no contexto do **ProcessoDeNegócio** que está sendo modelado. Uma **TarefaComposta** pode ser o recipiente de zero ou mais **TarefaSimples**, zero ou mais **ConjuntoDeEntradas** e um ou mais **ConjuntoDeSaídas.** Contem zero ou mais **DependênciaDeDados** através das quais conecta suas **Entradas** e **Saídas** com as **Entradas** e **Saídas** das **TarefaSimples** das quais é recipiente. Tem associada a máquina de estado ilustrada na Figura 3.2. Quando uma **TarefaComposta** é instanciada, cria instâncias de todas as **TarefaSimples** que contem (**ConjuntoDe-Dados, Dados,** e **DependênciaDeDados**). Quando um dos seus **ConjuntoDeEntradas** estiver habilitado, entra no estado 'executando-se' e seus valores podem ser consumidos pelas **TarefaSimples** que contem, de conformidade com as **DependênciaDeDados.** Entra no estado 'completado' quando nenhuma das **TarefaSimples** que contem estão no estado 'executando-se' e não existem **DependênciaDeDados** no processo de entrega de dados. Quando uma **TarefaComposta** entra no estado 'completado' seu **ConjuntoDeSaídas** fica disponível.

3.3.11 PapelNoProcesso

PapelNoProcesso define um lugar para colocar as entidades que num processo de workflow realizam uma **TarefaSimples** ou que são usadas (as entidades que a **TarefaSimples**

precisa para fazer o seu trabalho) na realização de uma **TarefaSimples.** Define um lugar para definir um comportamento. O proprietário de um **PapelNoProcesso** é uma **TarefaComposta** e o comportamento de **PapelNoProcesso** é parte do comportamento das **TarefaSimples** com as quais está associado. Contem dois atributos 'RegrasDeSeleção' e 'tipo' . 'RegrasDeSeleção' são expressões para buscar os objetos a ser enlaçados no momento de execução e 'tipo' determina o tipo das entidades compatíveis que podem ser ligadas quando a 'RegraDeSeleção' é avaliada. Quando uma **TarefaSimples** for instanciada, o atributo 'RegraDeSeleção' é avaliado para determinar a entidade que satisfaz a restrição expressada por ele. Este atributo pode se referir aos valores dos **Dados** pertencentes às **TarefaSimples** associadas com o **PapelNoProcesso,** ou a outros atributos das entidades associadas com **TarefaSimples,** via as associações tipo **realizadaPor**, **ParteResponsável** e/ou **usaArtefato.**

3.3.12 Executor

Executor como especialização de **PapelNoProcesso** é usado especificamente para identificar uma entidade que possa executar a **TarefaSimples** com a qual está associado (realiza o trabalho da **TarefaSimples**). Está contido em **TarefaComposta** e está associado com **TarefaSimples** via a associação **realizadaPor**

3.3.13 Artefato

Artefato como especialização de **PapelNoProcesso** é usado especificamente para identificar uma entidade que é requisitada como recurso por uma **TarefaSimples** (é um recurso usado pelo **Executor**). Está contido em **TarefaComposta** e está associado com **TarefaSimples** via a associação **usaArtefato.**

3.3.14 ProcessoDeNegócio

Estabelece o contexto no qual as atividades do negócio se realizam de maneira coordenada, para alcançar um objetivo de negócio. O processo de negócio existe no mundo real e está associado com uma definição de processo de workflow (**DefiniçãoDeWorkflow**) a qual define a sua parte automatizável (role **defineParteAutomatizavel**).

3.3.15 Publicador

O **Publicador** como uma especialização de **DeclaradorDeEventos** (ver seção 3.3.18) define a capacidade e interesse da **Tarefa** em produzir **NotíciasDeEvento** (ver seção 3.3.20) O **Publicador** está associado com uma ou mais **NotíciasDeEvento** e uma mesma **NotíciaDeEvento** pode estar associada com mais de um **Publicador**. Uma **Tarefa** é o recipiente de zero ou mais declarações de publicações de eventos.

3.3.16 Subscritor

O **Subscritor** como uma especialização de **DeclaradorDeEventos** declara a capacidade e interesse da **Tarefa** em consumir **NotíciasDeEvento.** O **Subscritor** está associado com uma ou mais **NotíciasDeEvento** e uma mesma **NotíciaDeEvento** pode estar associada com mais de um **Subscritor**. Uma **Tarefa** é o recipiente de zero ou mais declarações de subscrições de eventos.

3.3.17 **ParteResponsável**

ParteResponsável como uma possível especialização de **PapelNoProcesso** é usado especificamente para identificar uma entidade que tem a responsabilidade por uma **TarefaSimples** com a qual está associada via a associação **responsávelPor.**

3.3.18 **DeclaradorDeEventos**

O **DeclaradorDeEventos,** que esta relacionado com Tarefa e NoticiaDeEvento, modela de forma genérica os conceitos relacionados com eventos de workflow sem declarar se a **Tarefa** está interessada em produzi-los ou consumi-los. Tem como especializações **Publicador** e **Subscritor**.

Talvez a parte mais importante do metamodelo unificado proposto tenha a ver com a inclusão customizada dos conceitos de eventos provenientes do perfil EDOC, no contexto de workflow, já que eles estavam ausentes nos modelos de workflow analisados. A seguinte seção 3.3.19 detalhará a semântica dos eventos no contexto de workflow.

O modelo de eventos, ilustrado na parte direita da Figura 3.1, permite às tarefas (**Tarefa)** exporem suas ações ou mudanças de estado aos outros elementos. Um evento de workflow, explicitamente, expõe uma ação que tem uma semântica com significado para o sistema que está sendo modelado.

3.3.19 **EventoDeWorkflow**

Um evento de workflow, diferente de um evento de negócio, é a ocorrência de uma situação particular ou condição que seja de importância num (ou em vários) workflows. O conceito de evento de workflow permite ao sistema de workflow expor suas ações ou mudanças de estado a outros elementos do sistema permitindo-lhe a coordenação ou sincronização das diferentes atividades do sistema. A sua significância no contexto de um workflow pode ser :

- para iniciar ou terminar uma instância de workflow (ou, dito de uma maneira mais ge ral, para mudar seu estado);
- para permitir que uma atividade particular possa ser iniciada ou completada;
- para assinalar a outra instância de processo uma condição que será posteriormente usaa no processamento subsequente dessa instância de processo (ou de várias outras).

Um evento de workflow é composto conceitualmente por dois elementos: (i) um disparador da ocorrência e (ii) uma ação que responde a essa causa. Já que os eventos de workflow são visíveis no sistema de gerenciamento de workflow, é necessário introduzir os metaconceitos correspondentes aos disparadores de eventos e às ações dos eventos para identificar como os disparadores são conectados aos eventos e especificar as regras para interpretar o contexto do evento.

- disparador: é a causa e corresponde ao reconhecimento de um certo conjunto de circunstâncias associadas com a operação do sistema de workflow (ou alguma(s) de suas partes) e que ocasiona, como conseqüência, que seja realizada determinada ação. A semântica que corresponde ao reconhecimento do conjunto de circunstâncias associadas

(o disparador) está compreendida no metaconceito **RegraDeExposição** do metamodelo unificado- (ver seção 3.3.22). No contexto de workflow poderíamos distinguir dois tipos de disparadores: disparador relacionado com uma transição e disparador relacionado com uma condição de erro.

(i) Disparador relacionado com uma transição. Por exemplo, uma ação particular que ocorre numa **Tarefa** corresponde a fazer uma transição. Os estados associados com a máquina de estado de uma **TarefaSimples** e que representam uma transição poderiam ser usados na **RegraDeExposição** como disparadores assim: if estado = 'estado1' -> evento_nome (ver seção 3.3.22).

ii) Disparador relacionado com uma condição de erro. Por exemplo quando uma designação de recurso ou uma transição não pode ser feita.

- A ação: é a resposta representada pelo sistema de workflow (ou alguma(s) de suas partes) e que segue ao reconhecimento do conjunto de circunstâncias associadas com o disparador. As ações resultantes de um evento de workflow poderiam ser classificadas como:

  i) Explícitas, por exemplo definidas dentro da definição de processo de workflow para que o sistema de gerenciamento de workflow tome ações de controle específico como terminar uma instância de processo (em geral definir uma mudança de estado como resposta ao evento) ou como habilitar uma transição específica pendente mas esperando pela ocorrência de um evento de workflow. A semântica que corresponde ao reconhecimento do conjunto de circunstâncias associadas com a ação está compreendida no metaconceito **RegraDeNotificação** (ver seção 3.3.21) do metamodelo unificado. (ver na seção 3.5.7 as regras de comportamento CIMOSA, sugeridas para explicitar as **RegrasDeNotificação**).

  ii) Implícita (com respeito ao sistema de gerenciamento de workflow) a notificação a uma atividade particular ou instância de processo da ocorrência do evento. Corresponde ao caso em que uma instância de processo ou atividade requisita do Sistema de Gerenciamento de Workflow uma notificação de interesse de um tipo particular de evento. O sistema de gerenciamento do workflow não está interessado nos detalhes da ação, já que são responsabilidade da aplicação. Esse poderia ser o caso de uma entidade executora que, uma vez recebido o evento, saberá que ações particulares internas tomar.

Em síntese, um evento de workflow é aquele no qual o enlace entre a condição envolvida com o disparador e a ação[26] (resposta) resultante no sistema, envolve o sistema de gerenciamento de workflow. Em outras palavras, a ação é causada via o 'software' de gerencia-

[26] UML tem dois metaconceitos que em conjunto mapeiam a mesma semântica de evento: 'evento' e 'transição'. 'Evento' em UML é definido como o disparador ou causa à qual corresponderia o conceito de 'disparador' aqui introduzido, e 'Transição' em UML que é definida como a ocorrência de uma mudança de estado a qual mapeia uma possível conotação do conceito 'ação' acima introduzido no contexto de um evento de workflow. Em outras palavras, no contexto de workflow, o conceito proposto para um evento de workflow como formado por um disparador e uma ação corresponde respectivamente aos conceitos 'evento' e 'transição' do UML.

mento do workflow. Por exemplo, um evento poderia levar a que o sistema de gerenciamento de workflow habilite a iniciação de uma atividade em particular ou a terminação de uma instância de processo.

Um evento de workflow (**EventoDeWorkflow**) dispara uma ou mais notícias de evento (**NotíciaDeEvento)** –ver seção 3.3.20- e é descrito por (**descritoPor**) uma ou mais notícias de evento

3.3.19.1 Características dos eventos de workflow

Um evento de workflow pode ser caracterizado por: tipo, dados do evento e contexto.

tipo: Um evento de workflow pode ter associado um tipo para que o sistema de gerenciamento possa diferençar os eventos e suas correspondentes ações associadas.
A título de exemplo de diferentes tipos de evento e tomando como referência o modelo de interfaces que a OMG padronizou no seu Serviço para Gerenciamento de Workflow [3.2] podemos distinguir diferentes tipos de eventos de workflow, em particular para indicar que:

- um processo de workflow foi criado, o evento do tipo 'processoCriado' ('processCreated'),
- um estado do processo de workflow mudou, o evento do tipo 'estadoDoProcessoCambiado' ('processStateChanged')
- contexto do processo de workflow foi iniciado ou mudado , o evento do tipo 'contextoDoProcessoCambiado' ('processContextChanged),
- estado de uma atividade mudou, o evento do tipo 'CambiouEstadoDaAtividade' (activityStateChanged),
- contexto de uma atividade mudou , o evento do tipo 'CambiouContextoDaAtividade' ('activityContextChanged')
- resultado de uma atividade está pronto, o evento do tipo 'resultadoDaAtividadeCambiou' ('activityResultChanged')
- estado de uma designação de recurso mudou , o evento do tipo 'designação DaAtividadeCambiou' ('activityAssigmentChanged').

Dados do Evento: Um evento de workflow pode ter associado dados para permitir-lhe a transferência de mais informação da que pode ser derivada só do tipo do evento (desde o processo ou entidade que o emite até à entidade que o recebe). A meta-entidade do metamodelo unificado **NotíciaDoEvento** (ver seção 3.3.20) captura a semântica associada com os dados do evento. No caso anteriormente citado, os eventos do tipos 'estadoDoProcessoMudado' e 'estadoDaAtividadeMudado' podem agregar informação sobre a mudança de estado, em particular usando os atributos como 'estado_inicial' do tipo 'string' para registrar o estado anterior e o atributo 'novo_estado' também do tipo 'string' para registrar o novo estado. Os outros tipos de evento também podem agregar informação pertinente ao tipo. O Serviço de Gerenciamento de Workflow da OMG [3.2] especifica interfaces em IDL para registrar esses atributos e indagar sobre o valor deles.

Contexto: Um evento pode ter significado local só dentro de uma determinada máquina de workflow ou significado global através de várias máquinas de workflow que cooperam. Um aspecto importante do 'contexto' é definir o âmbito no qual os nomes dos tipos de evento de workflow são únicos.

3.3.20 **NotíciaDeEvento**

É a forma de modelar o anúncio de uma estrutura de dados relacionada e disparada por um evento de workflow (**EventoDeWorkflow)**. Uma notícia de evento é disparada por (**dispa-radoPor**) exatamente um evento de workflow (**EventoDeWorkflow)** e pode descrever no máximo um evento de workflow.

3.3.21 **RegraDeNotificação**

A semântica que corresponde ao reconhecimento do conjunto de circunstâncias associadas com uma ação que é resposta a um evento de workflow e que devem se satisfazer para a ação acontecer está compreendida no metaconceito **RegraDeNotificação** do metamodelo unificado. (ver na seção 3.5.7 as regras de comportamento CIMOSA, sugeridas para explicitar as **RegrasDeNotificação**).

3.3.22 **RegraDeExposição**

A semântica que corresponde ao reconhecimento do conjunto de circunstâncias associadas ao disparador de um evento de workflow e que devem se satisfazer para ele ocorrer está compreendida no metaconceito **RegraDeExposição** do metamodelo unificado.
É uma expressão booleana que especifica as condições ou conjunto de circunstâncias associadas com o disparador de um **EventoDeWorkflow** e que devem ser satisfeitas para que o evento de workflow seja produzido. Os termos usados na expressão booleana podem ser os atributos da entidade com a qual está associado o evento de workflow ou os atributos de outros elementos associados ou conteúdos dentro da entidade (no caso de uma **Tarefa** são seus **Dados**, **ConjuntoDeDados** e/ou **EventosDeWorkflow** contidos nela).

Podem previamente definir-se alguns nomes- padrão a serem usados como termos da regra de exposição e que causam a avaliação dela. Por exemplo, a palavra '*estado*' de tipo '*string'* com o significado de *satisfeito (estado = satisfeito)* no caso de um **ConjuntoDeEntradas** quer informar que está completo (if estado = satisfeito -> evento_nome). Para o caso de uma **Tarefa** que tem associada a máquina de estado ilustrada na Figura 3.2, podem definir-se também, como palavra padrão, '*estado*' e indicar que uma transição foi feita em sua máquina de estado como: estado = executando-se, o que ocasionaria a avaliação da regra de exposição (if estado = executando-se -> evento_nome). Conceitualmente para cada ação que aconteça no elemento fonte, a regra de exposição é avaliada e, se for verdadeira, o evento é construído e emitido.
Em geral, para efeitos de padronização, poder-se-ia usar, para declarar estas condições, a OCL 'Object Constraint Language' [3.1] .

3.3.23 **CondiçãoDeEvento**

As **RegraDeNotificação** podem ser restringidas, opcional e adicionalmente, por condições sobre eventos. A semântica da **CondiçãoDeEvento** permite exigir a chegada de eventos adicionais para que a regra de notificação possa ser avaliada em função deles (ver na seção

3.5.7 as regras de comportamento CIMOSA, sugeridas para explicitar as **RegrasDeNotificação).**

A Tabela 3.1 mostra a total coerência do metamodelo unificado proposto com o perfil EDOC, com exceção da semântica incluída no metaconceito **RegraDeExposição** que faz parte do metamodelo unificado e que não tem análogo no metamodelo de EDOC.

Tabela 3.1 Mapeamento entre os metaconceitos dos metamodelos unificado e EDOC.

DO METAMODELO UNIFICADO	DO PERFIL EDOC (Perfis para processos de negócio e eventos)
ConjuntoDeEntrada	GrupoDeEntrada(InputGroup)
ConjuntoDeSaída	GrupoDeSaída (OutputGroup)
ConjuntoDeDados	PortaMultipleDeProcesso(ProcessMultiPort)
Entradas	ConectorDePortaDeProcessoDeEntrada (input ProcessPortConnector)
Saídas	ConectorDePortaDeProcessoDeSaída (output ProcessPortConnector)
Dados	ConectorDePortaDeProcesso (ProcessPortConnector) / PortaDeFluxoDeProcesso (ProcessFlowPort)
Dependência DeDados	FluxoDeDados (DataFlow)
Tarefa	ComponenteDeProcesso (ProcessComponent)
TarefaSimples	Atividade (Activity)
TarefaComposta	TarefaComposta (CompoundTask)
ProcessoDeNegócio	ProcessoDeNegócio (BusinessProcess)
PapelNoProcesso	PapelNoProcesso (ProcessRole)
Executor	Executor (Performer)
Artefato	Artefato (Artifact)
ParteResponsável	ParteResponsável (ResponsibleParty)
Publicador	Publicador (Publisher)
Subscritor	Subscritor (Subscriber)
DeclaradorDeEventos	ProcessoBaseadoEmEventos (EventBasedProcess)
NotíciaDeEvento	NotíciaDeEvento (EventNotice)
EventoDeWorkflow	EventosDeProcesso (ProcessEvents)
RegraDeNotificação	RegrasDeNotificação (NotificationRules)

CondiçãoDeEvento	**CondiçãoDeEvento (EventCondition)**
RegraDeExposição	**Não está definido**

O mapeamento entre os metaconceitos do metamodelo unificado com o perfil de EDOC é praticamente direto, no que tem a ver com os conceitos relacionados com eventos. Os conceitos relacionados com eventos foram tomados diretamente do perfil de eventos para sistemas EDOC, guardando os relacionamentos entre eles e incluídos com os mesmos nomes no metamodelo unificado, com exceção só de **DeclaradorDeEventos** e **EventosDeWorkflow** que foram redefinidos, e a inclusão do metaconceito **RegraDeExposição**. O primeiro deles, para permitir especificar em geral a possibilidade de uma **tarefa** declarar a emissão ou subscrição de eventos, através do supertipo **DeclaradorDeEventos**, que é especializado pelas meta-entidades **Publicador** e **Subscritor**. Este supertipo **DeclaradorDeEventos,** na estrutura do metamodelo unificado, tem, com respeito à meta-entidade **tarefa,** uma simetria similar à que tem o supertipo **ConjuntosDeDados** que declara a potencialidade da **tarefa** ter conjuntos de dados e que é especializada como **ConjuntoDeEntradas** e **ConjuntoDe Saída**.

Um análise comparativa da Figura 3.1 com a Figura 2.3 do capítulo 2 mostra que, na Figura 3.1 do metamodelo unificado, os metaconceitos **ConjuntoDeEntradas** e **ConjuntoDeSaídas** (que são análogos aos metaconceitos do metamodelo da Nortel), foram desligados do metaconceito de *Evento,* ao qual estão ligados no metamodelo da Nortel. No metamodelo unificado especializam o metaconceito **ConjuntoDeDados** e as **Entradas** e **Saídas** são subtipos do metaconceito **Dados**.

3.4 MAPEAMENTO DO MODELO UNIFICADO AO UML

A UML possibilita, através do Grafo de Atividades [3.3], modelar os aspectos de controle e fluxo de dados dos processos de negócio [3.4]. O metamodelo unificado apresentado na seção 3.3 introduziu os metaconceitos necessários para descrever um processo de workflow consistente com os metaconceitos introduzidos no perfil EDOC.

Pelas razões enunciadas no capítulo 2, seção 2.3.3.1 "Uso não apropriado da semântica do UML no modelo EDOC", o perfil de UML para o metamodelo unificado será derivado do Grafo de Atividades de UML. O uso do Diagrama de Atividades de UML permite a modelagem de um processo de workflow de forma genérica, sem ter que determinar antes se o controle do processo será implementado de maneira centralizada ou distribuída, como não é o caso quando se deriva da metaclasse 'Colaboração' de UML como já foi exposto na seção 2.3.3.1. Em outras palavras, a semântica do Grafo de Atividades permite modelar os estados de um processo independentemente de que o controle (das ações) seja centralizado ou distribuído. Cada passo do processo é modelado como um estado do processo (o estado de realizar esse passo). Estes estados estão formalizados na máquina de estado associada com uma **Tarefa** (ver seção 3.3.8). As **restrições** sobre as **transições** entre os estados do processo permitem especificar a **ordem** e as **condições** sobre as quais os passos que compõem o processo são realizados.

O enfoque anterior não implica um determinado tipo de implementação como por exemplo em termos de mensagens passadas entre os passos ou entre os atores (**PapelNoProcesso**) responsáveis por estes passos (a designação deles aos passos do processo é suportada no Grafo de Atividade de UML através do conceito 'Partições de UML'). Em concreto existem diferênças semânticas substanciais entre o uso dos diagramas de colaboração e o uso do diagrama de atividades para modelar um processo de workflow.

A seguir, descreve-se como o metamodelo unificado pode ser definido como um perfil de UML 1.4, que estende o Grafo de Atividades de UML para modelar os aspectos de fluxo e de controle de dados e estende o metaconceito **DeclaradorDeEventos** da metaclasse 'Dependência' de UML (Core: Dependência) para estabelecer o relacionamento entre uma **Tarefa** (derivada da metaclasse de UML StateMachine::State) produtora ou consumidora de eventos e o conteúdo (**NotíciaDeEvento)** de um **EventoDeWorkflow** que é derivado da metaclasse de UML 'Estado de Fluxo do objeto' (StateMachine::ObjectFlowState). O **DeclaradorDeEventos** é derivado da metaclasse de UML 'Dependência', dado que ela pode relacionar qualquer dos elementos de modelagem de UML.

No metamodelo unificado, a semântica de uma ação particular está representada por **TarefaSimples** e um conjunto delas por **TarefaComposta**. Isto corresponde, no Grafo de Atividades (Activity Graph [27]) de UML, em geral, a Atividades (Activities) e em particular a suas especializações 'Estados de Ação' (ActionStates [28]) para as **TarefaSimples** e 'Estados de Sub Atividades' (SubActivityStates [29]) para **TarefaComposta**.

No metamodelo unificado, as **Tarefas** consomem dados (**Entradas**) e produzem dados (**Saídas**). Esta semântica pode ser representada nos Grafos de Atividades de UML através de um 'Estado de Fluxo do Objeto' (ObjectFlowState [30]) que no metamodelo de UML possuem uma referência a um 'Classificador' (Classifier) que representaria o tipo do dado.

No metamodelo unificado, **DependênciaDeDados** propaga os valores de dados entre as **Saídas** de uma **Tarefa** relacionada com as **Entradas** de outra. No Grafo de Atividades esta semântica pode ser representada, definindo um 'Estado de Fluxo do Objeto' que é a saída de uma Atividade e a entrada para outra. Assim **DependênciaDeDados** pode ser expressa-

[27] É um caso especial de máquina de estado que define um processo computacional em termos dos fluxos de controle e fluxos de objetos entre as ações que o constituem.

[28] É um estado que representa a execução de uma ação atômica, tipicamente a invocação de uma operação. A transição de saída é disparada pelo evento implícito de completar a execução da ação de entrada.

[29] Representa a execução de uma seqüência não atômica de passos. É um 'UML submachine state' que executa um Grafo de Atividades aninhado.

[30] Um objeto pode iniciar uma ação ou ser o objeto de uma ação. Um Fluxo do Objeto (ObjectFlow) define o fluxo de um objeto entre as ações num Grafo de Atividades. Um objeto de fluxo pode ser a entrada ou a saída de uma ação. Está associado com um 'classificador' (UML classifier) e com 'parâmetro' (UML parameter) que lhe provê as entradas e saídas. O mesmo objeto de fluxo pode ser manipulado por várias ações sucessivas e se encontrar depois delas em diferentes estados , dando lugar ao 'Estado de Fluxo do Objeto' (ObjectFlowState)

do como um estereótipo << **DependênciaDeDados >>** do conceito de Transição de UML com a seguinte semântica adicional de:

- conectar **Entradas** e **Saídas**
- permitir que lhe sejam agregadas as condições lógicas de pré- e pos-_ativação para que o fluxo de controle seja ativado
- uma vez que uma **Saída** esteja habilitada (ver seção 3.3.5), o valor dela passa através de <<**DependênciaDeDados** >> e fica disponível no 'Estado de Fluxo do Objeto', com o qual está conectado.

No metamodelo unificado as **Tarefas** contêm conjuntos de dados de entrada e de saída (**ConjuntoDeEntrada**/**ConjuntoDeSaída)** que agem como recipientes de dados (**Dados)** e como correlacionadores das dependências entre dados (**DependênciasDeDados)**. No Grafo de Atividades de UML os conjuntos de dados de entrada podem ser representados por um 'PseudoEstado de união' (join PseudoState [31]) e os de saída por um 'PseudoEstado de bifurcação' (fork PseudoState [32]). Vários conjuntos de dados de entrada ou de saída pertencentes a uma mesma tarefa podem ser modelados através de vários 'PseudoEstados de união e bifurcação'. A semântica desses 'PseudoEstados' é a mesma dos conjuntos de entrada e de saída, como expressado nas seções 3.3.1 e 3.3.2; em particular um **ConjuntoDeEntradas** é satisfeito quando todos seus valores forem recebidos (semântica AND). A Figura 3.3 ilustra o mapeamento acima descrito.

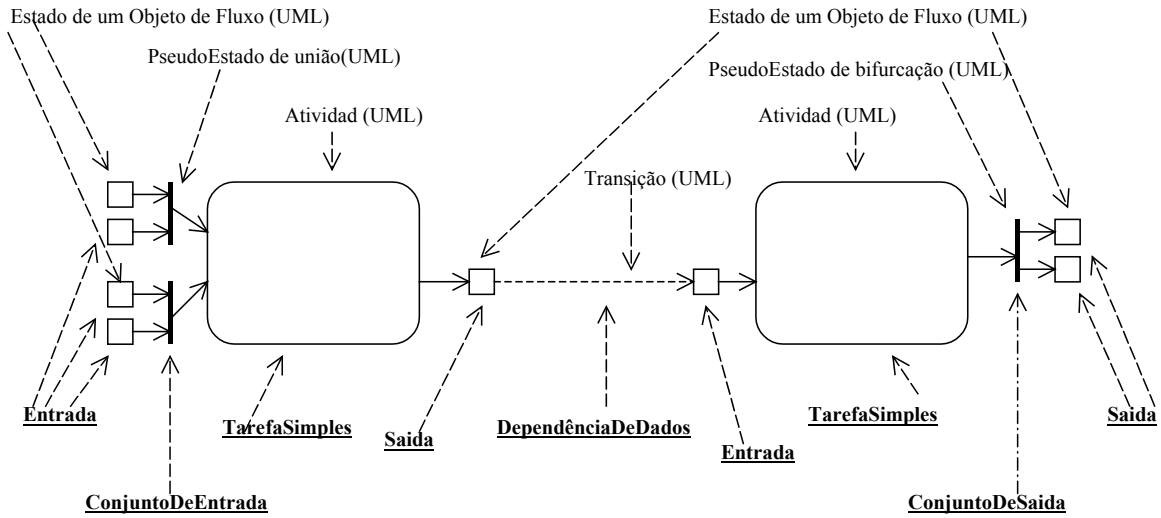


Figura 3.3. Representação de alguns metaconceitos do metamodelo unificado usando o Grafo de Atividades de UML.

[31] É um vértice no grafo da máquina de estado que serve para juntar várias transições que emergem de diferentes vértices fonte.

[32] É um vértice no grafo da máquina de estado que serve para separar uma transição de entrada em dois ou mais vértices destino

As decisões que representam as condições lógicas de pré- e pós- ativação, para que o fluxo de controle seja ativado, podem ser representadas em UML usando um 'PseudoEstado de decisão' (decision PseudoState) e colocando sobre a transição a condição a ser satisfeita. (ver na seção 3.5.5.1, dep(2) e dep(3))

Os **EventosDeWorkflow** podem ser modelados no contexto do Grafo de Atividades de UML também através do 'Estado de Fluxo do Objeto (StateMachine::ObjectFlowState) e o classificador associado com ele é um Sinal de UML (signal [33]) representando os atributos da **NotíciaDeEvento** (ver Figura 3.4)

Na Tabela 3.2 é apresentado o relacionamento entre os conceitos do metamodelo unificado representados por estereótipos de UML, e as correspondentes metaclasses de UML que serviram de base para cria-los. A semântica de cada estereótipo é a explicitada na seção 3.3 para cada uma das entidades que fazem parte do metamodelo unificado.

Tabela 3.2 O metamodelo unificado como estereótipo de UML 1.4

Estereotipo	Metaclasse em UML	Comentários
<< **TarefaSimples>>**	Estado de Ação	Com a semântica dada no MMU ; é recipiente de um ou varios <<ConjuntosDeEntrada>> e <<ConjuntosDeSaída>>
<< **TarefaComposta>>**	Estado de Sub Atividades	Com a semântica dada no MMU
<< **DependênciaDeDados>>**	Transição	Com a semântica dada no MMU ;faz parte de uma <<TarefaComposta>>
<< **ConjuntoDeEntrada>>**	PseudoEstado de união	Com a semântica dada no MMU; agrupa as <<Entradas>>
<< **ConjuntoDeSaída>>**	PseudoEstado de bifurcação	Com a semântica do MMU-agrupa as <<Saídas>>
<<**Entrada**>>	Estado do Objeto de Fluxo	Com a semântica dada no MMU, o tipo do valor de **Entrada** é representado pelo tipo do classificador associado com Estado do Objeto de Fluxo
<<**Saída**>>	Estado do Objeto de Fluxo	Com a semântica dada no MMU, o tipo do valor de

[33] É a especificação duma comunicação entre instâncias (um estímulo assíncrono comunicado). Um 'sinal' está associada com as características do comportamento que a originou. Uma sinal é definida independentemente das classes que operam com a sinal e esta especificada por uma máquina de estado. A recepção é uma declaração de que a classe manipulou a sinal.

		Saída é representado pelo tipo do classificador associado com Estado do Objeto de Fluxo
<<**PapelNoProcesso**>>	Partição	ver seção 3.4.1
<<**Executor**>>	Partição	ver seção 3.4.1
<<**Artefato**>>	Partição	ver seção 3.4.1
<<**ParteResponsável**>>	Partição	ver seção 3.4.1
<<**EventoDeWorkflow**>>	Estado de Fluxo do Objeto/Signal	
<<**DeclaradorDeEventos**>>	Dependência	
<<**NotíciaDeEventos**>>	Classificador	

A Figura 3.4 ilustra a derivação dos metaconceitos do metamodelo unificado a partir das metaclasses de UML 1.4. Na parte superior da figura estão as metaclasses de UML que servem como base da derivação e na parte inferior é representado o perfil de UML para o metamodelo unificado.

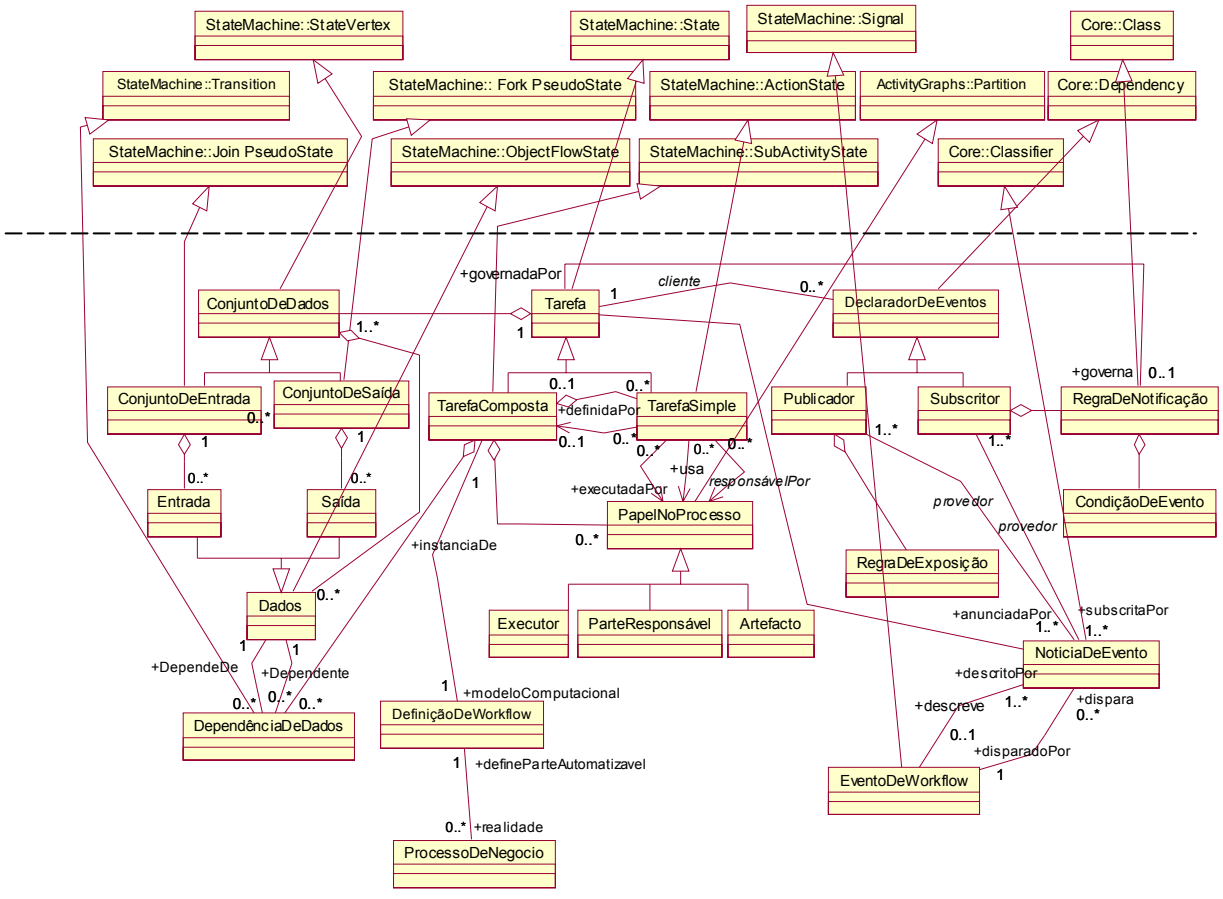


Figura 3.4 O metamodelo unificado como um perfil de UML 1.4

O fato de derivar o metamodelo unificado a partir do Grafo de Atividades de UML obriga a alguns ajustes na semântica de **PapelNoProcesso** do metamodelo de EDOC, os quais são descritos no parágrafo seguinte.

3.4.1 Implicações para EDOC da derivação do metamodelo unificado do grafo de atividades de UML

O uso semântico do grafo de atividades de UML para derivar o metamodelo unificado, como é ilustrado na Figura 3.4, cria dificuldades para associar uma **TarefaSimples** com seu potencial executor. A associação de uma **TarefaSimples** com a entidade **PapelNoProcesso**, via Partições de UML, como é ilustrado na Figura 3.4, obriga a ajustar a semântica definida em EDOC para a entidade **PapelNoProcesso**.

Especificamente, se o **PapelNoProcesso,** como identificador da entidade que realiza ou que é usada na realização de uma **TarefaSimples**, é considerado como uma Partição de UML então não é possível associar-lhe a semântica de EDOC, relacionada com os atributos 'RegrasDeSeleção' e 'tipo' já que a partição não é uma classe.

Se se fosse parcialmente consistente com EDOC, então **PapelNoProcesso** especializaria uma metaclasse de UML. Nesse caso, não seria possível, no metamodelo da Figura 3.4, associar **PapelNoProcesso** com o passo do processo (**TarefaSimples)**, já que (**TarefaSimples)** na Figura 3.4 deriva de 'Estado de Ação' (ActionState de UML) e 'Estado de Ação' no metamodelo de UML não está relacionado com outras metaclasses de UML.

Já que a metaclasse 'Dependência' de UML permite associar quaisquer dois tipos de elementos de modelagem de UML, uma saída está em derivar **PapelNoProcesso** da metaclasse 'Dependência' de UML e fazer com que os metaconceitos de **Executor** e **Artefacto** não sejam especializações de **PapelNoProcesso,** (como foi estabelecido no perfil EDOC) e sejam especificados melhor na forma de recursos independentes que derivem da metaclasse, classe de UML (UML Class).Desta maneira **PapelNoProcesso**, como 'Dependência' de UML, poderia associar, através de novas associações do tipo 'recurso_executor' e 'recurso_usado', os recursos **Executor** e **Artefacto** (derivados agora da metaclasse 'classe' de UML) com **TarefaSimples** (derivado de Estado de Ação de UML) e não se faria uso do conceito 'Partição' oferecido pelo Grafo de Atividade de UML. A seguinte Figura 3.5 ilustra essas alterações.

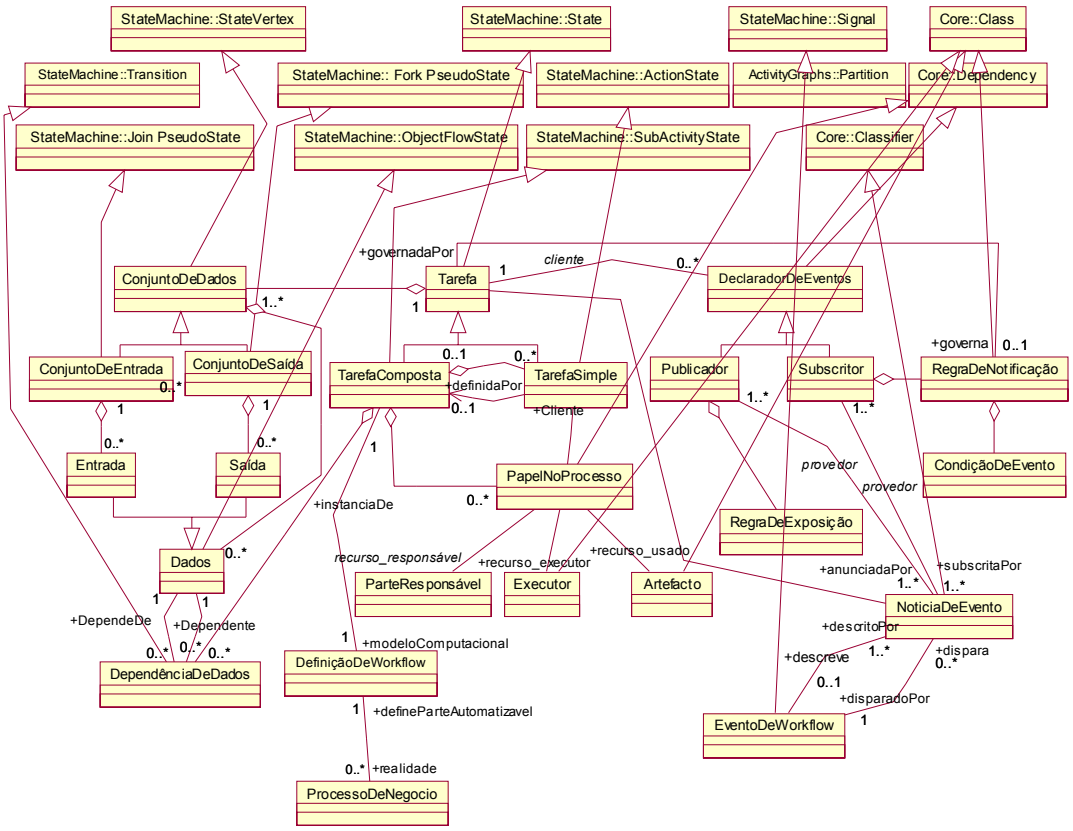


Figura 3.5 **O metamodelo unificado com o PapelNoProcesso como uma especialização da metaclasse Dependência de UML**

Em particular, deve notar-se na Figura 3.5 que **PapelNoProcesso** especializa a metaclasse Dependência de UML (e não Partições de UML) e que **Executor** e **Artefacto** são agora metaclasses de UML e não especializações de **PapelNoProcesso ,**o que permite a **PapelNoProcesso,** como dependência, enlaçá-las com as **TarefasSimples.**

Em resumo, apresentam-se três alternativas para tornar compatíveis o perfil de EDOC e o perfil do metamodelo unificado: (i) ajusta-se a semântica, definida por EDOC, da meta-entidade **PapelNoProcesso** para poder fazer uso do conceito de Partição de UML; (ii) altera-se a classe base utilizada por EDOC para estereotipar **PapelNoProcesso** para Dependência de UML ou (iii) ser totalmente compatível com o perfil de EDOC e usa-se a semântica de Colaboração de UML (UML Collaboration) para derivar o metamodelo unificado. Esta última alternativa obrigaria a deixar de lado a, bem aceita, forma de modelar processos de negócio (e seus associados processos de workflow) usando os diagramas de atividades, além de restringir-se a generalidade do modelo, como foi exposto na seção 2.3.3 do capítulo 2.

3.5 A LINGUAGEM TEXTUAL

Com base no metamodelo unificado descrito na secão 3.2 e 3.3 precedentes, foi desenvolvida uma representação textual que permite descrever o controle do fluxo do processo, tanto com base em dados, quanto em eventos, através da análise de regras de comportamento.

Após o processo de workflow ser modelado graficamente usando os metaconceitos que fazem parte do metamodelo unificado proposto na sessão 3.2 e do mapeamento apresentado na sessão 3.4, supõe-se a existência de um mecanismo, cuja execução faça a análise da representação gráfica do processo de workflow e gere automaticamente um esquema de workflow, num formato textual, de conformidade com o que será proposto mais adiante. Essa representação textual da definição de processo de workflow é necessária para que a definição possa ser reconhecida, interpretada e executada pela máquina de workflow.

Seguindo a convenção anteriormente enunciada para facilitar a diferenciação, comparação e identificação do contexto de procedência dos metaconceitos enunciados, os que provêm do metamodelo unificado serão escritos em **negritos sublinhados**, e as correspondentes expressões da linguagem textual serão escritas em letras 'tipo Arial' .

A linguagem proposta é composta por um conjunto de expressões e símbolos usados para especificar uma definição de processos de workflow, conforme com o metamodelo unificado proposto na seção 3.2 e 3.3. De acordo com o metamodelo unificado, um processo de negócio (**ProcessoDeNegócio)**, como metaconceito que represena a realidade, é definido em sua parte automatizável através de uma definição de workflow, que é instanciada através de uma tarefa composta (**TarefaComposta)** que esta associada com a definição de workflow. A tarefa composta (**TarefaComposta)** é constituída de tarefas mais simples (**TarefaSimples)** que representam as ações do processo de negócio. Tanto o processo (a **TarefaComposta**) como as tarefas simples (**TarefaSimples)** (as ações) que o constituem, contêm conjuntos de dados de entrada (**ConjuntoDeEntrada**) necessários para a execução e produzem conjuntos de dados de saída (**ConjuntoDeSaída**) como resultado da execução. Igualmente elas podem declarar a potencialidade de produzir (**Publicador**) eventos de workflow (**EventoDeWorkflow**) ou de consumi-los (**Subscritor**). As tarefas simples (**TarefaSimples)** podem ser vistas como definindo unidades atômicas de execução e as tarefas compostas (**TarefaComposta**) como sub-processos que agrupam, de maneira lógica, um conjunto de tarefas simples (**TarefaSimples),** dentro de outro processo de maior nível hierárquico. Desta maneira, o processo de maior nível hierárquico também pode ser considerado como uma tarefa composta (**TarefaComposta)**. Uma tarefa simples (**TarefaSimples)** pode ser descrita através de uma tarefa composta (**TarefaComposta**) via a associação **DefinidaPor** permitindo, mediante esta associação, especificar a utilização recursiva de subprocessos.

No metamodelo unificado, as tarefas simples (**TarefaSimples)** podem precisar de recursos especiais (**Artefatos**) para realizar seu trabalho, ou ser diretamente executadas por outras entidades (**Executores**) que satisfaçam as restrições explicitadas na meta-entidade **PapelNoProcesso** via a associação **executadaPor** ou **usa.** A meta-entidade **PapelNoProcesso** permite fazer a conexão entre o processo de workflow e as entidades necessárias para realizar cada passo do processo. O proprietário de um **PapelNoProcesso** é uma tarefa composta

(**TarefaComposta)** e o comportamento de **PapelNoProcesso** é parte do comportamento das tarefas simples (**TarefaSimpless)** com as quais está associado.

As dependências entre as tarefas (**Tarefa)** as quais determinam a sequência de execução das tarefas, podem definir-se de duas formas diferentes:

i) especificadas pela dependência (**DependênciaDeDados**) entre os dados produzidos (**Saídas**) por uma **Tarefa** e os dados de entrada (**Entradas**) que são necessários para execução de outra **Tarefa**, ou

ii) especificadas através de regras de comportamento que analisam eventos, usando **CondiçõesDeEvento** as quais permitem especificar as mudanças no processo que têm a ver com a entrada ou a saída de uma **Tarefa** . Estas declaração de condições do evento fazem parte do consumidor do evento (**Subscritor**).

A Figura 3.6 mostra um processo de workflow baseado no metamodelo unificado proposto, usando-se uma representação gráfica conforme com a proposta no perfil EDOC. A Figura 3.6 é seguida da modelagem do mesmo processo(ver Figura 3.7), utilizando-se o mapeamento entre os elementos do metamodelo unificado proposto e sua representação em termos do diagrama de atividades de UML, de conformidade com o mapeamento apresentado na seção 3.4 anterior.

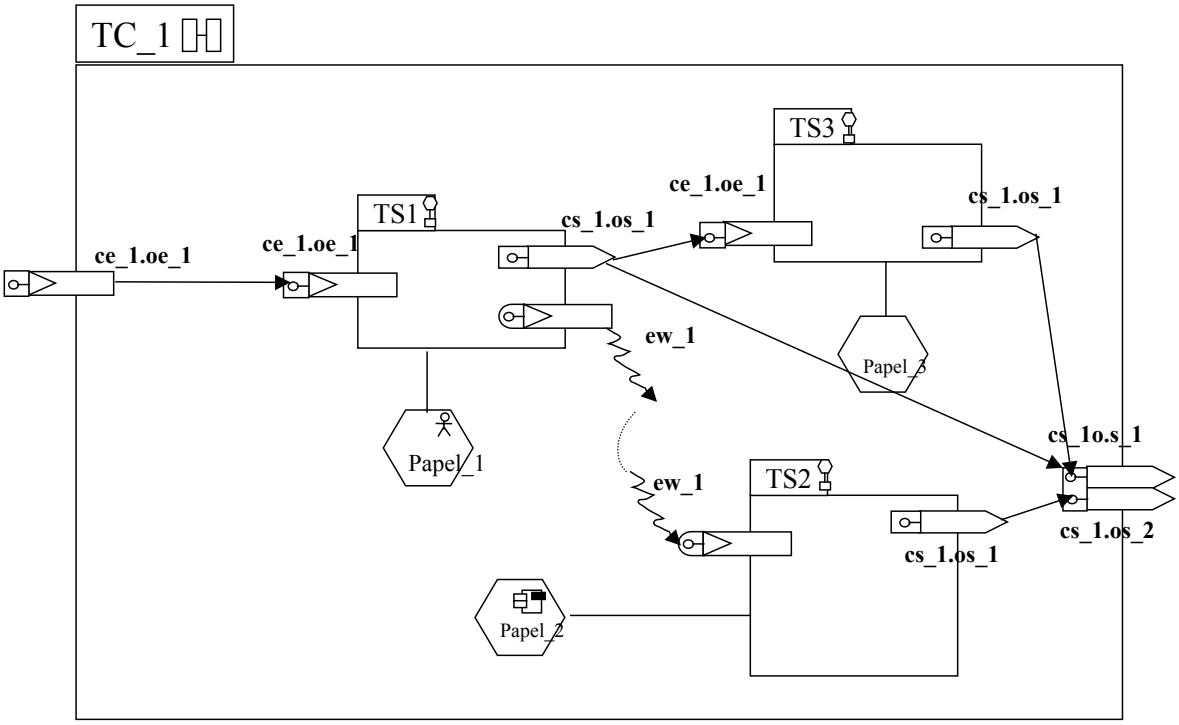


Figura 3.6 O Processo de Negócio TC_1, usando a notação gráfica própria do perfil EDOC

Nas Figuras 3.6 e 3.7 TC_1 é a **TarefaComposta** número 1, TS_1 a **TarefaSimples** número 1; ce_1.oe_1 o **ConjuntoDeEntrada** número 1 que contem a **Entrada** número 1 (o objeto de entrada número 1), cs_1.os_1 o **ConjuntoDeSaída** número 1 que contem a **Saída** número 1 (o objeto de saída número 1), e ew o **EventoDeWorkflow** número 1.

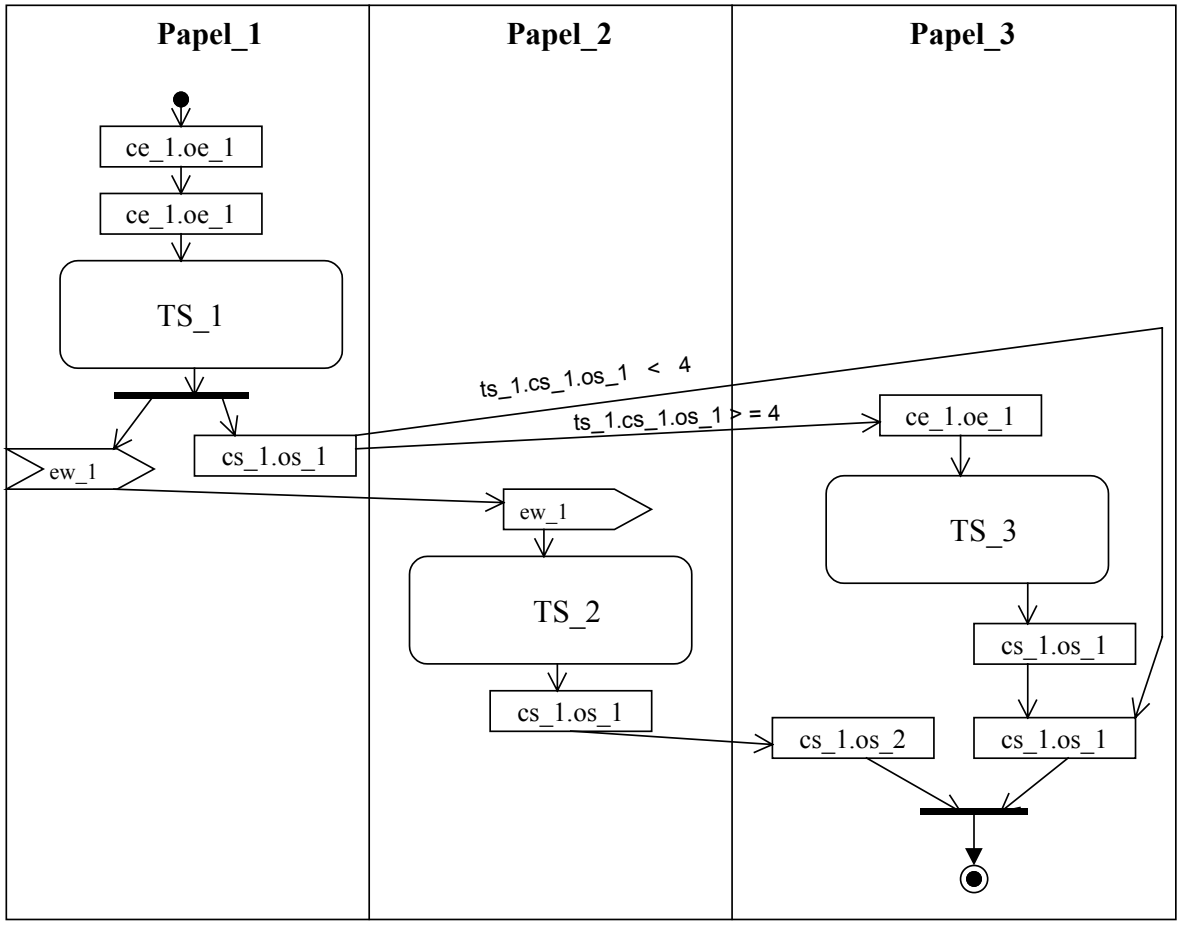


Figura 3.7 **Representação do Processo de Negócio TC_1, da Figura 3.6, usando o Diagrama de Atividades de UML**

A Linguagem está dividida em duas seções principais:

Numa primeira seção é declarada a informação que tem a ver com os tipos dos objetos de dados, dos eventos e dos executores que vão ser usados na definição do processo de workflow. Na segunda seção declara-se a informação que tem que ver com o processo de workflow propriamente dito. Além disso relacionam-se cada uma das tarefas que fazem parte da definição do processo de workflow. Dentro de cada tarefa são definidos seus conjuntos de entrada, os conjuntos de saída e os eventos a serem publicados e subscritos. Igualmente, definem-se a parte do fluxo de dados que especifica as dependências de dados entre cada uma das tarefas, a ordem de execução e as condições de pre- ativação e pós- ativação, a serem cumpridas em cada tarefa, para que uma transição seja realizada.

A título de exemplo, na Tabela 3.3 está representada a especificação correspondente ao processo modelado na Figura 3.7 anterior, usando a linguagem textual proposta. A Tabela 3.3 é seguida por uma explicação de cada uma das expressões introduzidas nela.

Tabela 3.3 **Definição textual do processo de workflow ilustrado na Figura 3.7**

```
//////////////////////////////////// Declaração de Dados////////////////////////////////////////////////////////////

struct Nome{
   int a;
   string b;
}

typedef sequence <Nome> NomeSequence;

///////////////Declaração de tipo de Eventos de workflow ////////////////////////////////////////

ew_1; MudouContextoDaAtividade;

/////////////// Declaração das Notícias de Evento////////////////////////////////////////////////////////////

ne_1{
   string nome;
   string estatura;
   int idade;
    (tipo nome;) *
};

ne_2{
   string tipo_de_evento;
   string Identificador_da_tarefa;
   string nome_da_tarefa;
   string identificador_da_tarefaComposta;
   string nome_da_tarefaComposta;
   string dado_inicial;
   string novo_dado;
}
//////////////////////////////////DefiniçãodoProcesso////////////////////////////////////////////////////////////

TarefaComposta( tc_1, “nome_do_processo”){
   ConjuntoDeEntrada( ce_1) {
        Entrada(oe_1, ' type_1' ); }
    ConjuntoDe Saída( cs_1 ) {
        Saída(os_1, ' type_3' ) ;
        Saída(os_2, ‘type_4’);}
    DefinidaPor() {
        TarefaSimples( ts_1, “nome_da_tarefa_1”, papel_1 ) {
                ConjuntoDeEntrada( ce_1) {
                        Entrada(oe_1, ' type_1' ); }
                ConjuntoDeSaída( cs_1 ) {
                        Saída(os_1, ' type_2' ); }
                Publicador( ew_1) {
```

```
            NotíciaDoEvento (ne_2); } }

      TarefaSimples( ts_2, "nome_da_tarefa_2", papel_2 ) {
            ConjuntoDe Saída( cs_1 ) {
                  Saída(os_1, ' type_4' ); }
            Subscritor( ew_1) {
            RegraDeNotification ( rn_1. 'condição -string literal'); } }

      TarefaSimples( ts_3, "nome_da_tarefa_3", papel_3 ) {
            ConjuntoDeEntrada( ce_1) {
                  Entrada(oe_1, ' type_2' ); }
            ConjuntoDeSaída( cs_1 ) {
            Saída(os_1, ' type_3' ); } }

  DependênciaDeDados (){
    dep( 1,  tc_1.ce_1.oe_1,  ts_1.ce_1.oe_1,  null,            null );
    dep( 2,  ts_1.cs_1.os_1,  ts_3.ce_1.oe_1,  ts_1.cs_1.os_1 > = 4, null );
    dep( 3,  ts_1.cs_1.os_1,  tc_1.cs_1.os_1,  ts_1.cs_1.os_1  <  4, null );
    dep( 4,  ts_2.cs_1.os_1,  tc_1.cs_1.os_2,  null,            null );
    dep( 5,  ts_3.cs_1.os_1,  tc_1.cs_1.os_1,  null,            null ); }
  } }
```

O resto desta seção descreve cada uma das expressões e símbolos usados na definição de processo de workflow da Tabela 3.3 anterior.

3.5.1 TarefaComposta

Uma tarefa composta serve para representar o processo de negócio como um todo e agrupa todas as unidades lógicas que representam os passos do processo de workflow. A sintaxe de uma tarefa composta deve possuir:

i) Uma ou mais seções para especificar os conjuntos de entrada (ConjuntoDeEntrada);
ii) uma ou mais seções para especificar os conjuntos de saída (ConjuntoDe Saída);
iii) uma seção para especificar as tarefas que a constituem (DefinidaPor)
iv) uma seção para especificar as dependências entre os dados que fazem parte das tarefas especificadas no literal anterior (DependênciaDeDados).

```
TarefaComposta (id, nome) {
    ....
}
```

O primeiro parâmetro 'id' é um identificador unívoco da tarefa composta (TarefaComposta) e o segundo parâmetro 'nome', permite fazer uma breve descrição do significado da tarefa composta.

3.5.2 TarefaSimples

Uma tarefa simples representa uma ação num processo de workflow. A sintaxe de uma tarefa simples pode possuir:

i) Uma ou mais seções para especificar seus conjuntos de entrada (ConjuntoDeEntrada);
ii) zero ou mais seções para especificar seus conjuntos de saída (ConjuntoDeSaída);
iii) zero ou mais seções para declarar sua intenção de gerar eventos (Publicador)
iv) zero ou mais seções para declarar sua intenção de consumir eventos (Subscritor)

```
TarefaSimples (id, nome, PapelNoProcesso) {
    ....
}
```

O primeiro parâmetro 'id' é um identificador unívoco da tarefa simples (TarefaSimples) , o segundo parâmetro 'nome', permite fazer uma breve descrição do significado da tarefa simples, o terceiro parâmetro ' PapelNoProcesso' é uma referência ou a uma entidade executora da tarefa, ou a um recurso especial que a tarefa possa precisar para realizar seu trabalho, ou ambas.

3.5.3 ConjuntoDeEntradas e ConjuntoDeSaída

Os conjuntos de Entrada (ConjuntoDeEntrada) e os conjuntos de saída (ConjuntosDeSaída) agem como correlacionadores dos objetos de dados de entrada (Entrada) e os objetos de dados de saída (Saídas) respectivamente.
A sintaxe dessas expressões é a seguinte:

```
ConjuntoDeEntrada( id ) {
  ...
  ...
  Entrada ( id, tipo );
  Entrada ( id, tipo );
  ...
}

ConjuntoDeSaída( id ) {
  ...
  ...
  Saída ( id, tipo );
  Saída ( id, tipo );
  ...
}
```

Com a expressões Entrada e Saída é representado o objeto de dados usado para transportar dados entre as tarefas. 'Entrada' representa o dado que entra e 'Saída' representa o objeto de dados que é produzido pela tarefa simples ou a tarefa composta que o contem

como parte de um grupo correlacionador de dados de saída. O primeiro parâmetro 'id' declara um identificador unívoco do objeto de dados, e o segundo parâmetro 'tipo' é utilizado para descrever qual é o tipo do objeto de dados, de conformidade com os tipos de dados declarados na seção respectiva.

3.5.4 DefinidaPor()

Com esta palavra chave define-se uma seção que faz parte só das TarefasCompostas e que serve para declarar as tarefas simples (ou tarefas compostas) que conformam a definição do processo de workflow, sem declarar suas interdependências. A sintaxe é a seguinte:

```
DefinidaPor() {
  ....
  ...
 TarefaSimples (id, nome, PapelNoProcesso) {
  ....
 }
}
```

3.5.5 DependênciaDeDados ()

Como já foi acima mencionado, as dependências entre as tarefas que determinam a sua sequência de execução podem ser definidas de duas formas diferentes :

i) especificada pela dependência entre os dados produzidos por uma tarefa (**Tarefa**) e os dados de entrada necessários para execução de outra tarefa, ou
ii) especificada através de eventos analisados por regras de comportamento, que têm a ver com as entradas ou saídas de uma **Tarefa** .

Em síntese, a execução de uma tarefa pode depender das Entradas/ ou Saídas de outras tarefas e/ou dos eventos publicados por outras tarefas.

3.5.5.1 Dependência de Dados

A expressão 'DependênciaDeDados' introduz uma seção que permite declarar a dependência entre os dados e corresponde ao caso mencionado no literal i) anterior. Esta seção (que só pertence às tarefas compostas), agrupa o registro de cada uma das relações de dependência individual entre as tarefas, usando a palavra chave 'dep' com quatro parâmetros. Cada "dep" dentro da seção 'DependênciaDeDados' mapeia o metaconceito **DependênciaDeDados** (fonte e destino) do metamodelo unificado e seus meta atributos pré- e pós-condição. As condições lógicas de saída, que a tarefa fonte deve cumprir, para que o fluxo de controle seja ativado, são declaradas, através de um dos parâmetros (o quarto parâmetro) com uma condição que chamaremos de condição de 'pós_ativação'. As condições lógicas que a tarefa destino deve cumprir para que o fluxo de controle seja ativado são declaradas através do quinto parâmetro, expressando outra condição que chamaremos de condição de 'pre_ativação'.

A sintaxes da seção DependênciaDeDados é a seguinte:

```
DependênciaDeDados ( ) {
  dep ( id, dependeDe, dependente, post_activação , pre_activação );

  dep( 2,  ts_1.cs_1.os_1,  ts_3.ce_1.oe_1,  ts_1.cs_1.os_1 > = 4, null );
  dep( 3,  ts_1.cs_1.os_1,  tc_1.cs_1.os_1,  ts_1.cs_1.os_1  <  4, null );
  ......
}
```

O significado de cada um dos cinco parâmetros da palavra chave 'dep' é o seguinte:

id: corresponde ao primeiro parâmetro de "dep" e declara um identificador unívoco da dependência individual entre cada par de tarefas.

dependeDe: corresponde ao segundo parâmetro de "dep" e identifica a referência ao objeto de dados do qual se depende e do qual faz parte da tarefa fonte e que corresponde também ao objeto de dados que vai ser transferido. Para garantir a unicidade deste identificador dentro da definição do processo de workflow, ele é declarado como constituído pelo identificador da tarefa fonte , pelo identificador do conjunto de dados e pelo identificador do objeto de dados, separados entre eles por pontos.. Por exemplo, ts1.cs1.s1, no primeiro campo 'ts1' representando a TarefaSimples com id = 1; no segundo campo o 'cs1' o identificador 'id' do conjunto de saída fonte (ConjuntoDeSaída(id)); e no último campo o identificador do objeto da dependência , a saída 's1'. Assim a dependência é desde o objeto de saída identificado como 's1', o qual faz parte do conjunto de saída identificado como 'cs1', e que pertence à tarefa simples identificada como 'ts1' .

Dependente: corresponde ao terceiro parâmetro de "dep" e identifica a referência ao objeto de dados a ser recebido na tarefa destino. Este identificador como o anterior é expressado em três campos separados por pontos: (i) o identificador da tarefa destino , (ii) o identificador do conjunto de dados de entrada e (iii) o identificador do objeto de dados. Por exemplo, o terceiro parâmetro tipo ts2.ce1.e1 se interpreta como, 'ts2' representando a TarefaSimples com id=2 como tarefa destino da dependência, 'ce1' representando o identificador 'id=1' do conjunto de entrada (ConjuntoDeEntrada(id)) que faz parte da tarefa destino, ts2; e finaliza com o identificador do objeto de dados final da dependência 'e1', representando a Entrada 1.

Pós_ativação: captura uma condição de pós-ativação na tarefa fonte que está identificada no segundo parâmetro da expressão "dep" . Deve cumprir-se essa pós-condição na tarefa fonte para que a transição se efetue, no caso de não estar esse parâmetro declarado como "null".

Pre_ativação: captura uma condição de pré-ativação na tarefa destino que está identificada no terceiro parâmetro da expressão "dep" . Deve cumprir- se essa pré-condição na tarefa destino para que a transição se efetue, no caso de não estar declarado esse parâmetro como "null".

As pré- e pós condições podem incluir referências a:

- saídas de dados de outras tarefas das quais se depende (por exemplo, que o objeto de dados da saída da tarefa fonte seja maior ou igual a 4, ‘os1 >= 4’). Isto permite especificar condições sobre as entradas alternativas de dados (Entradas) a uma tarefa. Por exemplo, é possível que, sob certas circunstâncias, a saída (s1) de uma tarefa ‘ts1’ seja a entrada (e1) de uma outra ‘ts2’, mas que noutra situação a entrada da tarefa ‘ts2’ seja alimentada por uma terceira tarefa ‘ts3’. Logo, existiriam duas dependências, chegando à tarefa ‘ts2’ com diferentes condições:

  dep (1, ts1.cs1.os1, ts2.ce1.oe1, os1 >= 4, null)
  dep (2, ts3.cs1.os1, ts2.ce1.oe1, os1 < 4, null),

  Assim se o objeto de dados de saída da tarefa fonte ts1 for maior que 4 ou igual aocorrerá a transição entre a tarefa ‘ts1’ e ‘ts2’, caso contrário, a transição será entre ‘ts3’ e ‘ts2’.
- eventos previamente declarados (ver seção 3.5.5.2 Dependência de Eventos).

3.5.5.2 Dependência de Eventos

O conceito de eventos, implícito na metaclasse abstrata **DeclaradorDeEventos** do metamodelo unificado, permite às tarefas simples e compostas expor suas ações ou mudanças de estado aos outros elementos subscritores do evento. Um evento de workflow (**EventoDeWorkflow**) explicitamente expõe uma ação que tem uma semântica com significado para o sistema que está sendo modelado (ver seções 3.5.6 e 3.5.7 seguintes).

3.5.6 Publicador / RegraDeExposição

A sintaxe para que a entidade **TarefaSimples**, a qual é representada na notação textual pela expressão TarefaSimples (ts_1, ,), declare seu interesse em ser um publicador do evento de workflow do tipo ‘ew_1’ é através da expressão :

```
Publicador( id) {
NotíciaDoEvento ( id);
RegraDeExposição ( id, ‘condição literal’);}
```

onde a palavra chave Publicador declara que a entidade vai publicar eventos, em particular o evento de workflow identificado através do parâmetro ‘id’, por exemplo Publicador(ew_1) para o caso de declarar a emissão de um evento do tipo ew_1, ou Publicador(MudouContextoDaAtividade) no caso de declarar a emissão do evento do tipo MudouContextoDaAtividade. O conteúdo do evento de workflow identificado como ’ew_1’ é referenciado como um parâmetro da palavra chave NotíciaDoEvento através do identificador ‘id’ da notícia do evento, por exemplo NotíciaDoEvento (ne_1), ou NotíciaDoEvento (ne_2). O identificador ‘ne_1’ referencia uma estrutura de dados associada com o

evento particular ‘ew_1’, e o identificador ‘ne_2’ uma estrutura de dados que poderia estar associada com o evento do tipo particular ‘MudouContextoDaAtividade’ (ver Tabela 3.3, e seção 3.5.6.1). Em outras palavras, existe um identificador para o evento e outro para o conteúdo dele.

Em síntese a estrutura de dados **NotíciaDoEvento** é disparada (sempre quando a RegraDeExposição, se for definida, tornar-se verdadeira) pelo **EventoDeWorkflow** e é comunicada pela infra estrutura,a partir dos publicadores **(P ublicador/** Publicador() **)** aos subscritores **(Subscritor/** Subscritor() **)** .

3.5.6.1 Especificação textual de Eventos

Os dados da (**NotíciaDoEvento/**NotíciaDoEvento**)** associados com os eventos (**EventoDeWorkflow)** são especificados usando uma sintaxe similar à usada na declaração de estruturas em IDL (Interface Definition Language), assim:

Nome { (type name;) * }

Onde ‘Nome’ é o identificador que aparece como parâmetro da palavra chave NotíciaDeEvento. Por exemplo, o caso do evento do tipo ‘MudouContextoDaAtividade’, mencionado na seção 3.3.19.1, pode ter associado como NotíciaDeEvento a estrutura de dados ne_2 seguinte:

```
ne_2{
      string tipo_de_evento;
      string Identificador_da_tarefa;
      string nome_da_tarefa;
      string identificador_da_tarefaComposta;
      string nome_da_tarefaComposta;
      string dado_inicial;
      struct novos_dado;
}
```

Assim:

```
Publicador(CambiouContextoDaAtividade) {
      NotíciaDoEvento ( ne_2);
      RegraDeExposição ( id, ‘condição literal’);
}
```

3.5.6.2 Especificação textual da RegraDeExposição

A regra de exposição é uma expressão booleana que especifica as condições sobre as quais um evento é produzido e é introduzida com a palavra chave RegraDeExposição e dois parâmetros: o primeiro parâmetro é um identificador da regra de exposição e o segundo é

um literal para descrever a condição que deve ser satisfeita para o evento ser produzido (ver seção 3.5.6.1).

Na notação textual proposta, a expressão que declara a potencialidade de gerar um evento de um determinado tipo, com um conteúdo dado por uma NotíciaDoEvento e que é guardado por uma dada RegraDeExposição, mantem a mesma estrutura (seqüência de palavras chave, parêntesis, colchetes, pontos e comas) usada na declaração das entradas (ConjuntoDeEntrada) e saídas (ConjuntoDeSaída) de uma tarefa (TarefaSimples) Assim:

```
ConjuntoDeSaída( ) {
    Saída(  , ' ' ); }
```

tem a mesma estrutura textual que:

```
Publicador ( ) {
    RegraDeExposição ( , ' ');}
```

3.5.7 Subscritor / RegraDeNotificação

A entidade **TarefaSimples**, representada na notação textual como TarefaSimples (id, .. , ..) declara seu interesse em ser um subscritor (**Subscritor**/ Subscritor) do evento de workflow do tipo 'id', com a expressão:

```
Subscritor( id) {
    RegraDeNotificação ( id, 'condição literal');}
```

na qual a palavra chave Subscritor indica que a entidade está interessada em escutar, receber, ou ser notificada dos eventos de workflow do tipo identificado através do parâmetro 'id'. Por exemplo Subscritor(ew_1), no caso de estar interessado em 'escutar' eventos de workflow do tipo 'ew_1'.

A regra de notificação, com a semântica definida em **RegraDeNotificação,** é representada na notação textual como RegraDeNotificação e determina o que irá acontecer numa entidade que possua um **DeclaradorDeEventos** do tipo **Subscritor,** quando a **NotíciaDoEvento** for recebida pelo processo que faz a subscrição ao evento de workflow. Uma regra de notificação é introduzida com a palavra chave RegraDeNotificação e dois parâmetros, o primeiro parâmetro é um identificador da regra, e o segundo uma condição de tipo literal**.** A condição é uma expressão que pode estar baseada nos atributos da meta-entidade **NotíciaDoEvento**, a qual é recebida pelo Subscritor na forma de uma estrutura de dados, de acordo com a declaração feita pelo Publicador através da expressão do tipo NotíciaDoEvento(ne_1). A condição descreve, mediante um subconjunto da estrutura ne_1, a mudança na entidade proprietária (o **Subscritor**) desta **RegraDeNotificação.** As mudanças na entidade proprietária têm a ver com as entradas ou saídas dos nós que representam os passos do processo de workflow (ou a saída de um nó e a posterior entrada num outro nó representando uma transição entre dois nós).

Nem o perfil EDOC, nem o modelo de workflow da Nortel (que só usa um tipo simples de eventos relacionado com a satisfação dos *ConjuntoDeEntradas (InputSet*) e os *ConjuntoDeSaída (OutputSet)* , nem mesmo o modelo de workflow da WfMC (que não tem nada a ver com eventos) especificam ou sugerem tipos de regras de notificação (**RegrasDeNotificação),** baseadas em eventos e associadas com o comportamento do processo. Com o objetivo de formalizar, na linguagem textual proposta, a especificação das regras de notificação relacionadas com eventos, sugere-se utilizar as regras de comportamento como foram definidas no projeto CIMOSA [3.5]

CIMOSA (Computer Integrated Manufacturing-Open System Architecture) é um esforço Europeu orientado à modelagem empresarial e à padronização da integração empresarial. A especificação em CIMOSA do comportamento de um processo é realizada por um conjunto de condições e de ações associadas com as condições, do tipo WHEN (condição) DO regra de ação (ver primeira coluna da Tabela 3.4).

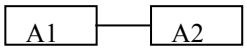
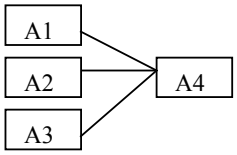
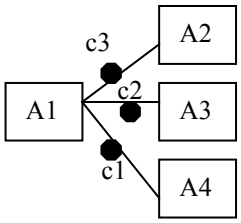
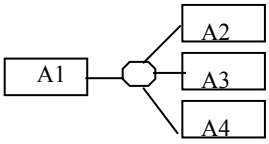
Na notação textual proposta, as regras de comportamento, como foram estabelecidas em CIMOSA, podem ser usadas para especificar o comportamento baseado em eventos de um processo de workflow. Em particular, podem ser usadas para especificar o segundo parâmetro ('condição literal') da expressão RegraDeNotificação**.** Essa condição, especificada no segundo parâmetro de RegraDeNotificação, define as mudanças no processo relacionadas com as entrada ou as saídas dos nós do processo de workflow.

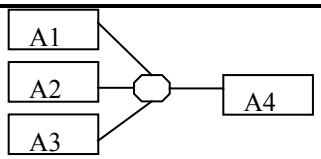
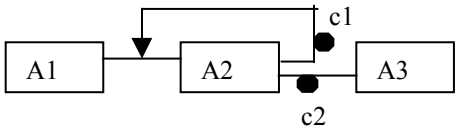
A potencialidade destas regras de notificação para suportar diferentes possibilidades de roteamento entre as tarefas de um processo de workflow, fica ilustrada na Tabela 3.4, com o mapeamento entre essas regras e as possibilidades de roteamento definidas no modelo de workflow da WfMC. A inclusão destas regras na linguagem, para definir processos de workflow, permite sua especificação baseada em eventos, com todas as possibilidades do roteamento clássico entre atividades, presente no modelo de workflow da WfMC. A inclusão destas regras permite a definição de um processo de workflow misto, no sentido de possibilitar a especificação das dependências, tanto por eventos quanto por dados.

A Tabela 3.4 apresenta uma síntese das regras de comportamento de CIMOSA e seu mapeamento com as possibilidades de roteamento entre atividades estabelecidas no modelo de workflow da WfMC [4.5]

Tabela 3.4 Mapeamento entre as regras de comportamento de CIMOSA e as possibilidades de roteamento entre atividades da WfMC

CIMOSA- Regra de Comportamento	WfMC- Possibilidade de roteamento
Regras para iniciar um processo: são usada para iniciar um processo de negócios PN1 com eventos.	Iniciação do processo

WHEN (START WITH event-1) DO PN1	
Regras de seqüência: são usadas para representar condições de roteamento no fluxo de controle. EF(PN1) é uma função que retorna o Estado Final EF( ) do processo de negócios PN1, ANY é uma palavra reservada que representa todos os possíveis estados do PNi, e estado-1, é um estado final especifico, WHEN (EF(PN1) = ANY) DO PN2 (regra de seqüência forcada) WHEN (EF(PN1) = estado-1) DO PN2 (regra de seqüência condicionada ao estado-1)	Caminho simples (single thread): uma atividade A2, pode ser executada depois da terminação de outra, A1  ou união (or-join): um ponto no workflow no qual dois ou mais ramais alternativos de atividades de workflow convergem a uma atividade comum como seguinte passo no workflow  ou bifurcação (or-split): um ponto no workflow no qual um único caminho de controle toma a decisão sobre que via tomar quando se encontra com vários caminhos. A decisão é especificada pelas condições de transição C1, C2 e C3, 
Regras de bifurcação: usadas para representar a execução paralela de PNi; '&' é o operador paralelo, e SYNC é uma palavra reservada significando o inicio simultâneo de PN2, PN3, PN4 WHEN (EF(PN1) = estado-1) DO PN2 & PN3 & PN4 (chamado bifurcação assíncrona) WHEN (EF(PN1) = estado-1) DO SYNC (PN2 & PN3 & PN4) (chamado bifurcação síncrona)	e- bifurcação (and split): um único caminho de controle se separa em dois ou mais caminhos os quais são executados em paralelo, permitindo que múltiplas atividades sejam executadas simultaneamente. 
Regras de união(Rendezvous) : são usadas para sincronizar a finalização de uma bifurcação. WHEN (EF(PN1) = estado-1 AND EF(PN2) = estado-2 AND EF(PN3) = estado-3) DO PN4	e- união (and-join): um ponto no qual dois ou mais atividades executando-se em paralelo convergem a um caminho (thread) simples e comum de controle.

	
Regras de repetição (loop): são usadas para executar o mesmo PNi iterativamente WHEN (EF(PN1) = valor da repetição) DO PN1	Iteração (Iteration): um ciclo que envolve a execução repetitiva de uma ou mais atividades de workflow até a satisfação de uma condição. C1 e C2 são condições de transição. 
Regras de terminação do processo: são usadas para indicar a finalização do processo. WHEN (EF(PN1) = estado-1) DO FINISH	Finalização do processo

O uso, na linguagem textual, das regras de comportamento de CIMOSA, para especificar o segundo parâmetro (‘condição literal’) da expressão RegraDeNotificação, permite o seqüenciamento das ações do processo de workflow, orientado por eventos, com um comportamento análogo às possibilidades de roteamento oferecidas no modelo de workflow da WfMC.

Conclusões do Capítulo

Este capítulo apresenta um metamodelo para representar a lógica de um processo de workflow consistente com o metamodelo do perfil EDOC. A consistência esta dada por o mapeamento apresentado entre os metamodelos que mostra que é possível expressar uma definição de processos de workflow e uma especificação funcional do sistema de suporte com termos (meta-conceitos) que são mapeáveis. O mapeamento assegura que não exista ambigüidade entre a forma como o analista de processos de negócio descreve os processos de workflow e a forma como o projetista do software de suporte descreve o sistema que suportara esse processo de negócio. Também é apresentada a derivação do metamodelo unificado como um perfil de UML o que permite ao metamodelo unificado ser representado usando os diferentes diagramas de UML em particular o Diagrama de Atividades de UML para representar a lógica de um processo de workflow.

3.6 REFERÊNCIAS DO CAPÍTULO

[3.1] UML 1.4 - Chapter 6 - Object Constraint Language Specification OMG Document Number: formal/01-09-77; 2001.
[http://ftp.omg.org/pub/docs/formal/01-09-77.pdf

[3.2] Juergen Boldt. Workflow Management Facility Specification (WMF), v1.2, OMG Document Number: formal/00-05-02; 2000.
[http://ftp.omg.org/pub/docs/formal/00-05-02.pdf]

[3.3] Editor, Ms. Linda Heaton. "Unified Modeling Language Specification v1.4 "; 2001, OMG Document Number: formal/01-09-67 a formal/01-09-80[http://ftp.omg.org/pub/docs/formal/01-09-67.pdf]

[3.4] Craig Dewalt, Business Process Modeling with UML. OMG Document:ad/00-02-04; 2000.

[3.5] CIMOSA. A Primer on key concepts, purpose and business values. http://cimosa.cnt.pl/Docs/Primer/primer0.htm , (Accessed : 22/01/20002)]

[3.6] Workflow Management Coalition. Terminology and Glossary. Document Number WfMC-TC-1011, v3.0 (Feb 99).

CAPÍTULO 4

ASPECTOS DE IMPLEMENTAÇÃO

4.0 INTRODUÇÃO

Nos capítulos anteriores foram expostos o marco conceitual e os resultados teóricos que sustentam os aspectos da implementação a serem apresentados neste capítulo. Em particular, foram descritos o metamodelo unificado para definir o processo de workflow e especificar o software de suporte, e uma notação textual plausível para representar os processos de workflow, coerente com o metamodelo unificado. Neste capítulo serão descritos alguns aspectos da implementação, derivada e apoiada nos resultados teóricos mencionados.

A funcionalidade básica dos mercados de serviço eletrônico [4.1] pode ser aumentada com a inclusão das funcionalidades da tecnologia de workflow . A inclusão de funcionalidades de workflow permite coordenar a execução de vários serviços, numa cadeia, aproveitando-se o fluxo da informação entre eles como mais um valor agregado.

A plataforma Platin [4.2], por oferecer um ambiente básico para a implementação de um mercado eletrônico aberto de serviços (embora sem funcionalidades de workflow), foi utilizada como ambiente de testes para a integração de uma máquina de workflow, baseada em [4.3], e que interpreta uma definição de processo de workflow consistente com a linguagem textual apresentada na seção 3.3.

Na seção 4.1 é apresentada uma breve síntese descrevendo o ambiente de integração –a Plataforma Platin-. A seção 4.2 apresenta o cenário da aplicação que serviu para modelar o processo de workflow, interpretado pela máquina de workflow integrada em Platin. A seção 4.3 apresenta a máquina de workflow implementada. A seção 4.4 ilustra os resultados da execução. A seção 4.5 detalha algumas das interações básicas entre a máquina de workflow e a Plataforma Platin. Na seção 4.6, são apresentadas as características do sistema e software usados e na seção 4.7 comentam-se os resultados da implementação.

4.1 A PLATAFORMA PLATIN

A plataforma Platin [4.2], baseada em CORBA [4.4], forma um ambiente de processamento para um mercado de serviços abertos [4.5] no qual podem ser administradas e executadas aplicações distribuídas. Este ambiente, previsto na Arquitetura de Redes de Informação de Telecomunicações (TINA-Telecomunications Information Networking Architecture) [4.6], é composto por nós de processamento distribuído, heterogêneos e interconectados (Distributed Process Environment nodes- DPE nodes), que basicamente respondem a dois propósitos: (i) esconder a heterogeneidade organizacional e técnica, e (ii) fornecer um sistema que facilite as tarefas dos desenvolvedores das aplicações distribuídas.

A Plataforma Platin ajuda a separar as aplicações dos sistemas necessários para suportá-las, de tal maneira que ambas as partes possam evoluir em linhas diferentes de tempo.

Na Plataforma podem distinguir-se três camadas diferentes: (i) Aplicações, (ii) DPE (Distributed Processing Environment) e (iii) NCCE (Native Computing and Communication Environment) (ver Figura 4.1).

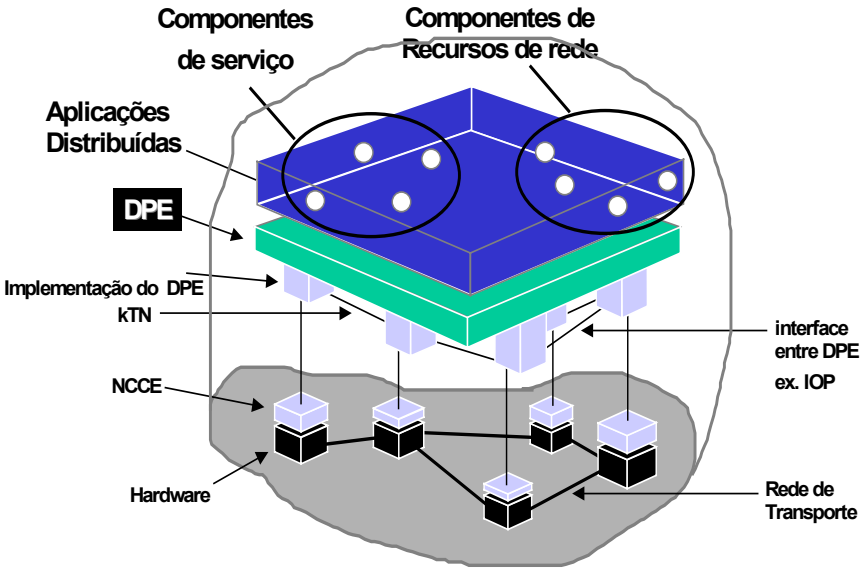


Figura 4.1 Diferentes camadas na Plataforma Platin [4.2]

A camada de Aplicação

Essa camada é composta por serviços de aplicação e de informação. Esses serviços são construídos como componentes de software e oferecem a funcionalidade aos usuários finais (pessoas) embora outras aplicações interessadas na mesma funcionalidade possam também ser usuários dela. Assim, interagir com uma determinada aplicação, consiste em usar o(s) serviço(s) que ela oferece. As aplicações consistem de objetos computacionais que requerem ser instalados antes de ser executados. Assim uma aplicação distribuída terá seus objetos computacionais distribuídos em diversos nós de processamento. Um dos propósitos da plataforma é permitir a instalação, execução e interação entre esses objetos computacionais.

Camada do Ambiente de Processamento Distribuído (DPE -Distributed Processing Environment*)*

Esta camada é constituída pelas capacidades computacionais para acessar e controlar a infra-estrutura e executar os componentes de software. O DPE é a infra-estrutura que o software das aplicações distribuídas usa como ambiente de execução. Essa execução é transparente com respeito à distribuição. O DPE é constituído por nós interconectados com o propósito de permitir a interoperação entre as aplicações e seus objetos computacionais instalados em diferentes nós.

Camada do Ambiente Nativo de Computação e Comunicação (NCCE - Native Computing and Communication Environment)

Esta camada é a parte baseada na tecnologia da rede e compreende os recursos da infra estrutura que permitem o transporte de dados. As aplicações que residem no DPE devem po-

der acessar a infra-estrutura de transporte da rede permitindo às aplicações usar e controlar suas capacidades de transporte.

Um mercado aberto de serviços [4.5], além de suportar aplicações distribuídas em geral, deve poder respeitar o relacionamento entre o usuário e o provedor de um serviço. A arquitetura de serviços de Platin está baseada na arquitetura de serviços de TINA [4.6] que descreve os componentes necessários para acessar e usar os serviços num mercado de serviços aberto. O que tipifica esses serviços é o fato deles se encontrarem em ambientes distribuídos. Companhias diferentes oferecem componentes implementados em diferentes computadores, usando linguagens diferentes, combinadas de forma a fornecer um serviço comum.

A arquitetura da plataforma em questão oferece um 'framework' que leva em consideração os requerimentos de um mercado de serviços abertos e que descreve a forma como os serviços são desenvolvidos, oferecidos e fornecidos.

O modelo de participação ilustrado na Figura 4.2 é derivado da arquitetura de serviços de Platin e representa os diferentes papeis dos participantes, num mercado aberto de serviços.

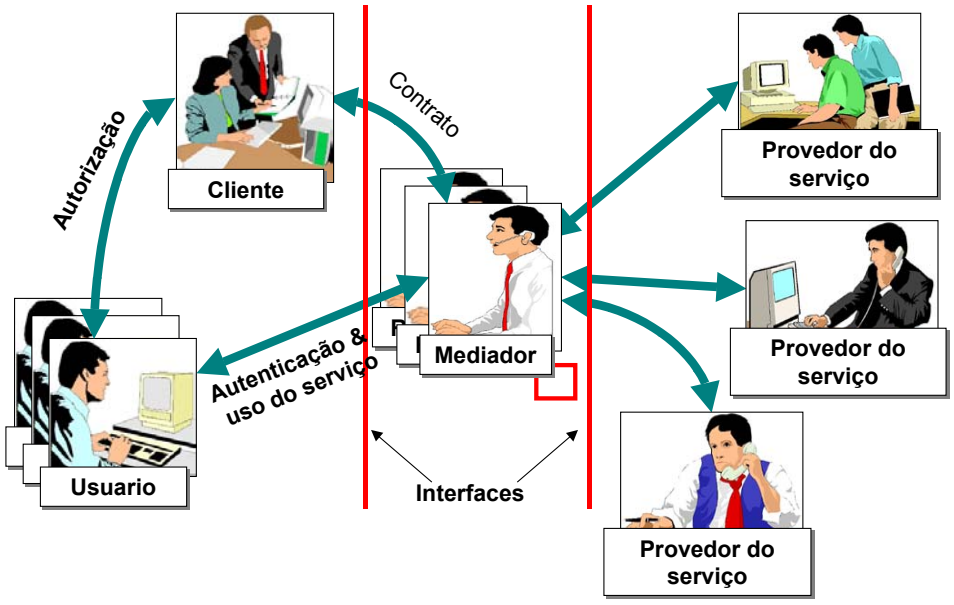


Figura 4.2 **Papeis no mercado de serviços implementado sobre a plataforma Platin [4.2]**

O mediador vende uma variedade de serviços aos clientes. O mediador e o cliente subscrevem um contrato especificando os acordos para o uso de um serviço. O cliente autoriza a seus próprios empregados (usuários) o uso do serviço subscrito. O mediador, através da plataforma, oferece várias funções:

Um ponto de acesso comum a todos os serviços que está mediando.

- A verificação da identidade do cliente e do usuário.
- Verificação dos direitos do usuário, tais como quais serviços podem ser usados por um determinado usuário.
- Controle da sessão de serviços (iniciar, suspender, terminar, etc.)

- Contabilidade dos custos, contraídos pelo usuário durante a sessão do serviço, em nome do provedor do serviço
- Administração de todos os clientes e serviços oferecidos (mediados) através da plataforma.

4.2 CENÁRIO DE APLICAÇÃO

Para ilustrar o uso da máquina de workflow implementada e integrada na plataforma Platin, será descrito o cenário de uma aplicação hipotética de viagens, oferecida como um serviço mediado através da plataforma Platin. A aplicação de viagens é constituída por dois serviços a serem encadeados pela máquina de workflow, (i) um Serviço de Reserva de Hotel e (ii) um Serviço de Reserva de Passagens, os quais têm uma dependência simples de dados entre eles. Em primeira instância, deve ser executado o serviço "ReservaDeHotel" e só depois dele ser executado com sucesso, seu resultado será transferido como dados de entrada ao segundo serviço "ReservaDePassagens", para que os dados resultantes do passo anterior sejam novamente processados. O resultado da execução desse segundo serviço corresponde ao resultado final da "AplicaçãoDeViagens". A combinação desses dois serviços, numa cadeia, aproveita o fluxo da informação como um valor agregado. A Figura 4.3 apresenta graficamente a lógica desse processo usando um diagrama de atividades de UML 1.4

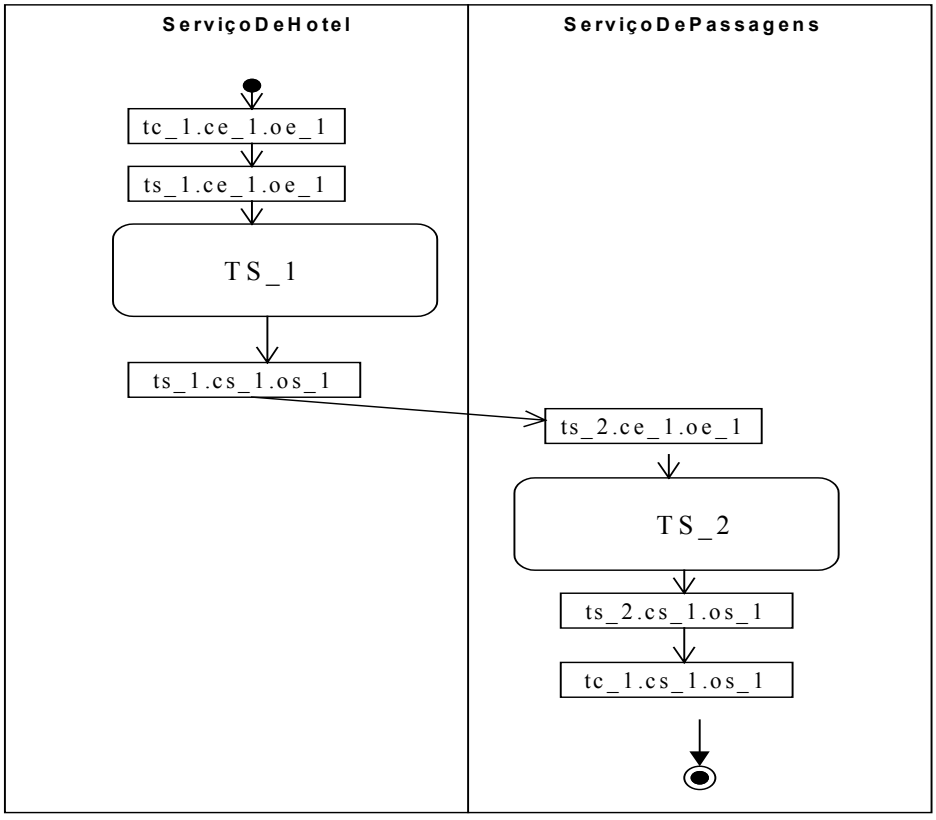


Figura 4.3 A lógica da Aplicação de Viagens como um diagrama de atividades de UML 1.4

A representação gráfica ilustrada na Figura 4.3 deve ser mapeada para a notação textual introduzida no capítulo 3 seção 3.4 para ser computacionalmente interpretável pela máquina de workflow. A Tabela 4.1 apresenta em notação textual a lógica da aplicação de viagens acima descrita. Na seção **DependênciaDeDados** o quarto e o quinto parâmetro que fazem parte da especificação correspondentes a cada uma das dependências (**dep**) não aparecem na representação textual por não ser suportados pela atual implementação da máquina de workflow.

Tabela 4.1 Representação textual da lógica do processo ilustrado na Figura 4.3

```
TarefaComposta(tc_1, "AplicaçãoDeViagens"){
  ConjuntoDeEntrada( ce_1) { Entrada( oe_1, 'd' ); }
  ConjuntoDeSaída(cs_1) { Saída( os_1, 'd' ) ; }
DefinidaPor() {
  TarefaSimples(ts_1,"ReservaDeHotel",ServiçoDeHotel) {
                    ConjuntoDeEntrada( ce_1 ) { Entrada( oe_1, 'd' ); }
                    ConjuntoDeSaída( cs_1 ) { Saída( os_1, 'd' ); } }
  TarefaSimples(ts_2,"ReservaDePassagens",ServiçoDePassagens) {
                    ConjuntoDeEntrada(ce_1) { Entrada( oe_1, 'd' ); }
                    ConjuntoDeSaída(cs_1) { Saída( os_1, 'd' ); } }
  DependênciaDeDados (){
       dep( 1, tc_1.ce_1.oe_1, ts_1.ce_1.oe_1 );
       dep( 2, ts_1.cs_1.os_1, ts_2.ce_1.oe_1 );
       dep( 3, ts_2.cs_1.os_1, tc_1.cs_1.os_1 );}}
  }
```

A interpretação desta definição de processo é a seguinte. A lógica da aplicação de viagens, como um todo, é introduzida usando a palavra-chave TarefaComposta. A palavra-chave DefinidaPor é usada para introduzir a descrição dos passos que fazem parte de todo o processo. Dentro da seção DefinidaPor cada um dos passos "ReservaDeHotel" e "ReservaDePassagens" é descrito usando os parâmetros da palavra-chave TarefaSimples. Tanto a TarefaComposta como a TarefaSimples especificam, por separado, suas respectivas informações de entrada e de saída, através das palavras-chave ConjuntoDeEntrada e ConjuntoDeSaída. Dentro de cada um deles, os objetos de dados, a serem realmente transferidos entre os passos do processo, são especificados usando as palavras-chave Entrada e Saída.

A seção introduzida com a palavra-chave DependênciaDeDados () descreve as dependências entre os passos que conformam o processo. Cada dependência é individualmente especificada usando a palavra-chave dep(). Assim dep (1,...) indica que o objeto de entrada (oe_1), que faz parte do conjunto de entrada 1,(ce_1) da tarefa composta 1 (tc_1) é, por sua vez, o objeto de entrada 1 (oe_1) que faz parte do conjunto de entrada 1(ce_1) da tarefa simples 1. Os detalhes da sintaxe associada com estas expressões podem ser vistos na seção 3.4 do capítulo 3 e os resultados da execução deste processo dentro da Plataforma Platin na seção 4.4 deste capítulo.

4.2.1 Especificação da aplicação de viagens

De conformidade com o perfil EDOC (ver capítulo2, Tabela 2.3), um **ProcessoDeNegócio (BusinessProcess)** é visto externamente como um **ComponenteDeProcesso (Process-Component)** que define uma unidade de comportamento configurável (software) e internamente como uma **TarefaComposta (CompoundTask)** que define a forma de coordenar um conjunto de **Atividades (Activities)** relacionadas. Os **ComponentesDeProcesso** (**ProcessComponent)** definem os pontos de interação entre eles através de portas conhecidas como **PortaDeFluxoDeProcesso (ProcessFlowPort)** as quais definem as interfaces e operações delas. O uso dessas portas numa Atividade é representada através do metaconceito **ConectorDePortaDeProcesso (ProcessPortConnector).** Os relacionamentos entre os **ConectoresDePortasDeProcesso (ProcessPortConnectors)** das atividades relacionadas é representado através de **FluxoDeDados (FluxoDeDados)** que também propagam valores de dados.

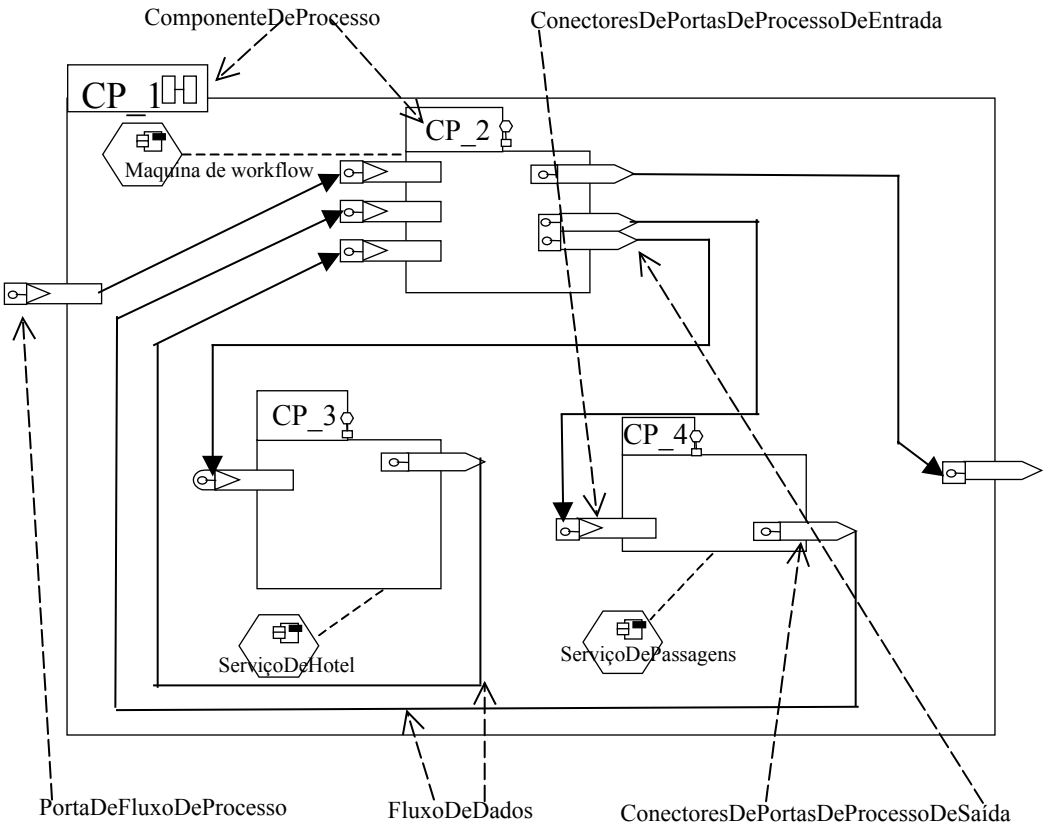


Figura 4.4 A Aplicação de Viagens como ComponenteDeProcesso

Usando a notação do perfil EDOC e no nível de detalhe mais externo a Figura 4.4 mostra a aplicação de viagens especificada como um **ComponenteDeProcesso** (CP_1) que inter-

namente é composta por outros três componentes de processo CP_2, CP_3 e CP4. Estes três componentes de processo correspondem à representação das **Atividades** de maior hierarquia (as mais externas) envolvidas na aplicação de viagens: (i) CP_2 à atividade correspondente com a lógica do processo. (ii) CP_3 à atividade relacionada com o serviço de Hotel e (iii) CP_4 atividade relacionada com o serviço de Passagem. Cada uma das atividades está relacionada com uma entidade executora: CP_2, representando a lógica do processo, é executada pela entidade chamada 'máquina de workflow', CP_3 e CP_4 executadas pelas entidades chamadas ServiçoDeHotel e ServiçoDePassagens respectivamente. Cada um dos **ComponentesDeProcesso** expõe suas portas para iniciar ou responder aos fluxos de dados entre eles. A atividade CP_2 expõe três **ConectoresDePortasDeProcesso (ProcessPortConnectors)** na entrada e três na saída. As atividades CP_3 e CP_4 expõem só um **ConectorDePortasDeProcesso (ProcessPortConnectors)** na entrada e outro na saída. Os três **ConectoresDePortasDeProcesso** na entrada de CP_2 visam (i) receber a informação de contexto de todo o processo, (ii) os resultados da realização da atividade ServiçoDeHotel e (iii) os resultados da realização da atividade ServiçoDePassagens. Os três **ConectoresDePortasDeProcesso (ProcessPortConnectors)** saída de CP_2 visam (i) controlar a realização de CP_3, (ii) controlar a realização de CP_4 e (iii) entregar os resultados da realização da lógica do processo ao **ComponenteDeProcesso** mais externo que representa à aplicação de viagens como um todo. Os **ConectoresDePortasDeProcesso (ProcessPortConnectors)** nas entradas e saídas de CP_3 e CP_4 visam oferecer uma porta para serem controlados por CP_2 e outra para entregar seus resultados. O relacionamento entre os diferentes **ConectoresDePortasDeProcesso (ProcessPortConnectors)** é chamado **FluxoDeDados (DataFlow)** e ilustra o fluxo de informação entre eles.

Usando os metaconceitos que fazem parte do metamodelo unificado (ver capítulo 3, Tabela 3.1) a aplicação de viagens pode ser igualmente descrita usando o metaconceito 'Tarefa' no lugar de **ComponenteDeProcesso**, 'Entradas' no lugar de **ConectoresDePortasDeProcessoDeEntrada** 'Saídas' invés de **ConectoresDePortasDeProcessoDeSaída** e 'DependênciaDeDados' por **FluxoDeDados**.

A definição do processo do workflow, a ser interpretado pela máquina de workflow, e que corresponde à lógica da aplicação de viagens, usando os metaconceitos do metamodelo unificado, é ilustrada na Figura 4.3 e Tabela 4.1 anteriormente apresentada.

4.3 A MÁQUINA DE WORKFLOW

A Figura 4.5 ilustra o diagrama de classes de UML correspondente à máquina de workflow implementada e integrada na Plataforma Platin.

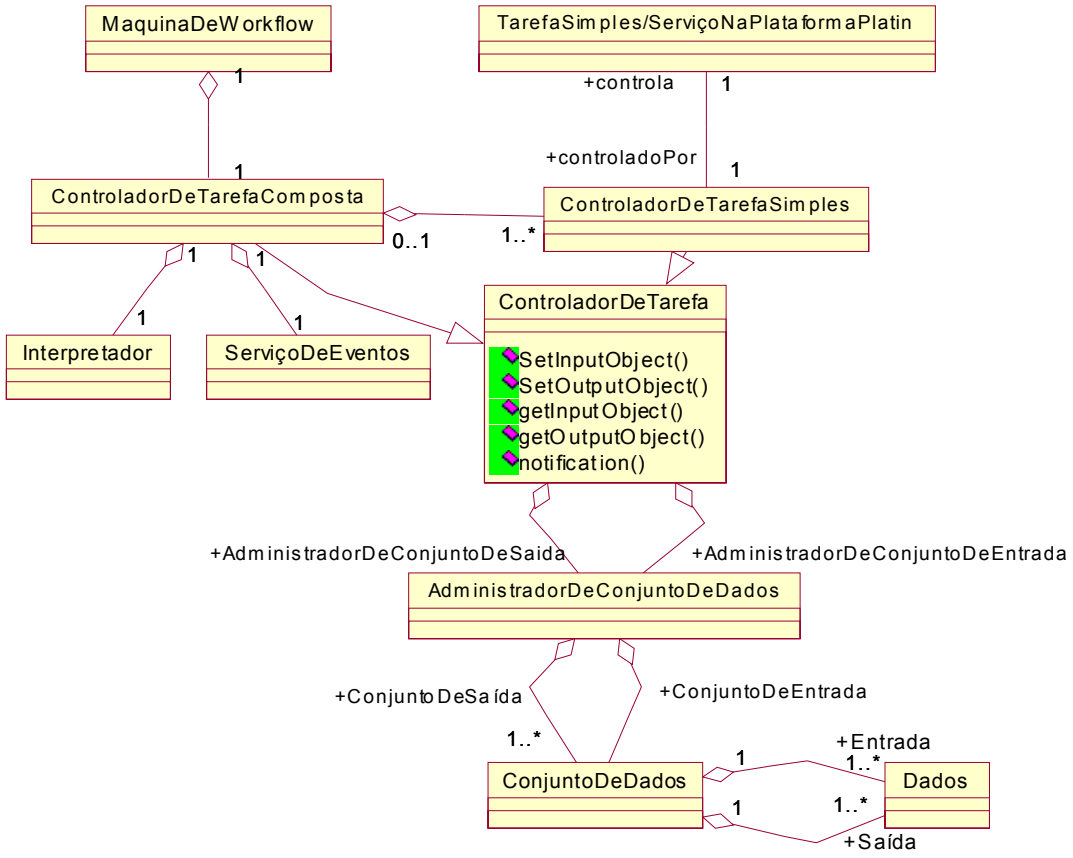


Figura 4.5 Diagrama de classes da máquina de workflow

A seguir é descrita, de forma simplificada, a estrutura das classes que implementam a máquina de workflow.

```
ControladorDeTarefa
    atributos
        String id; //identificador do controlador
        String state; //estado associado com a tarefa sendo controlada
    Operações
        String get_id();
        String get_state();
        int SetInputObject ( String id, DataObject object ); //para definir o valor do objeto de Dados
        int SetOutputObject( String id, DataObject object );
        int getInputObject( String id, DataObject object ); //para obter o valor de objeto de Dados
        int getOutputObject( String id, DataObject returned_object );
        int notification( Vector event_list);
```

O ControladorDeTarefa é uma classe abstrata que define um conjunto de métodos e atributos para serem reutilizados entre os seus herdeiros os ControladorDeTarefaComposta e o ControladorDeTarefaSimples. A principal função do ControladorDeTarefa é receber notificações (método int notification(...)) sobre as dependências que foram satisfeitas e, se estes eventos o habilitarem, disparar a execução das entidades executoras. Os métodos que com-

põem esta classe permitem definir (int SetInputObject(..) e int SetOutputObject(..)) e recuperar (int getInputObject(...) e int getOutputObject (...)) os objetos de Dados.

```
ControladorDeTarefaComposta
    atributos
        ServiçoDeEventos serviçoDeEventos ;
        Vector ControladoresDeTarefas; //referências aos controladores
        Interpretador DefiniçãoDoProcesso; //interpretador da definição de processo
    Operações
        SetDependency ( Interpretador DefiniçãoDoProcesso); //para inserir as dependências
        void complete (Vector ControladoresDeTarefas); //para notificar a todos os
        //ControladoresSimples a finalização do processo
```

O ControladorDeTarefaComposta agrega o Interpretador de definições de processo de workflow e o ServiçoDeEventos. É responsável por instanciar todos os ControladoresDe TarefaSimples que contém (Vector ControladoresDeTarefas) e definir as dependências entre eles (SetDependency(...)) de acordo com a definição de processo de workflow interpretada através do Interpretador.

```
ControladorDeTarefaSimples
    atributos
        String id; //identificador do controlador
        String state;
        struct context_data;
        struct result_data;
    Operações
        String get_id();
        String get_state();
        int SetInputObject ( String id, DataObject object ); //para definir o valor do objeto de Dados
        int SetOutputObject( String id, DataObject object );
        void start();
        set context (struct[] new_value);
        struct[] result();
        void receive_call_back( struct[] result);
```

A principal função ControladorDeTarefa Simples é receber as notificações propagadas pelo ServiçoDeEventos , verificar a habilitação de algum de seus ConjuntosDeDados e, se for habilitado, iniciar e ativar a entidade que realmente realizará o trabalho (no caso o serviço na Plataforma Platin). Quando esta produzir os resultados deve-se atribuí-los ao ConjuntoDeSaída adequado.

Interpretador

É a classe que interpreta a definição do processo de workflow a nome do ControladorDeTarefaComposta e instância de acordo com ela os ControladoresDeTarefasSimples e as dependências entre eles

```
ServiçoDeEventos
    atributos
        ControladorDeTarefaComposta parent; //referência ao controlador que o contem
```

```
            Vector simple_task_list; //referência a todos os ControladoresDeTarefaSimples;
            AdministradorDeConjuntoDeDados ConjuntoDeSaída;
    Operações
            void broadcast (Vector event_list, ControladorDeTarefaComposta owner);
```

Quando um ConjuntoDeEntrada ou um ConjuntoDeSaída de qualquer ControladorDeTarefa é completado, é através desta classe que são notificados (Vector simple_task_list) os demais ControladoresdeTarefaSimples

```
Dados (DataObject)
    atributos
            String id; // identificador do objeto
            String state; //estado do objeto- recebeu ou não um valor
            struct valor data; // estrutura com a informacao a ser transferida
            Vector Events;
            AdministradorDeConjuntoDeDados owner;
    Operações
            setDataObject (DataObject object);
            DataObject getDataObject ();
            String getId();
```

É a classe que modela em geral os dados (os objetos de dado). Uma instância dela transporta dados entre os diferentes controladores de tarefa.

```
ConjuntoDeDados
    atributos
            String id;
            String state;
            Vector data_objects;
            Vector Events;
            AdministradorDeConjuntoDeDados owner;
    Operações
            void becomeSatisfied ();
            int notification ( Vector event_list);
            int setInputObject( String id, Dado src_obj, ControladorDeTarefaSimples owner);
            int setOutputObject( String id, Dado src_obj, ControladorDeTarefaSimples owner);
            int getInputObject( String id, Dado ret_obj);
            int getOutputObject(String id, Dado ret_obj);
            void insertEvent ( String s);
```

É a classe que possui a informação sobre os dados (Vector Dados), dependências (Vector Dependências) e eventos (Vector Eventos) que um ControladorDeTarefas necessita para exercer seu controle (void insertEvent (..); SetInputObject (...); getInputObject(...), etc).

```
AdministradorDeConjuntoDeDados
    atributos
            Vector setVector // vector de ConjuntoDeDados que administra;
            boolean satisfied;
    Operações
            ConjuntoDeDados findSet ();
            int getOutputObject( String id, DataObject obj);
```

```
int setOutputObject (String id, DataObject src_obj, ControladorDeTarefa owner);
int setInputObject (String id, DataObject src_obj, ControladorDeTarefa owner);
int getInputObject( String id, DataObject ret_obj);
```

A principal função do AdministradorDeConjuntoDeDados é prover métodos para definir, recuperar os conjuntos de dados e os dados.

MáquinaDeWorkflow

A máquina de workflow agrega o ControladorDeTarefaComposta. Em nível de execução, a máquina de workflow, quando interpreta a definição do processo, instancia um ControladorDeTarefaComposta (tc_1.java na Figura 4.6) para toda a Aplicação de Viagem e, para cada um dos passos que conformam o processo (ReservaDeHotel e ReservaDePassagem) um ControladorDeTarefaSimples (ts_1.java e ts_2.java na Figura 4.6).

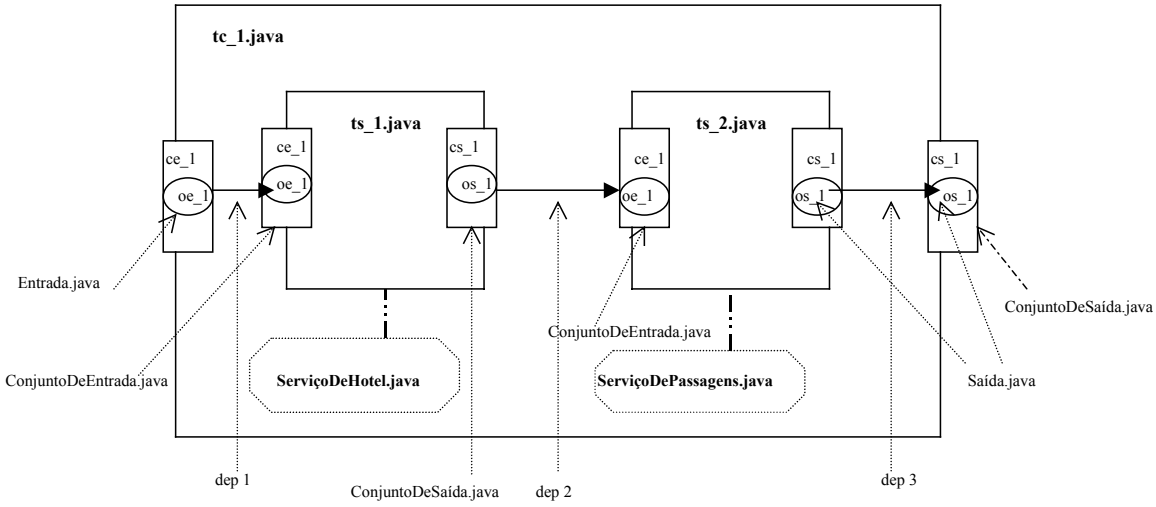


Figura 4.6 A aplicação de viagem e as classes Java que a implementam

Quando a informação de entrada está disponível no ControladorDeTarefaComposta (tc_1.java na Figura 4.6) uma notificação é enviada (usando o ServiçoDeEventos que ControladorDeTarefaComposta possui, ver Figura 4.4) ao ControladorDeTarefaSimples (ts_1.java), que controla o passo relacionado com o serviço de "ReservaDeHotel" anunciando-lhe que sua informação de entrada está disponível para ser utilizada (corresponde à dep 1 na Figura 4.6 e dependência 1, na Tabela 4.1). Esse ControladorDeTarefaSimples requer da entidade executora associada com ele (na Figura 4.6, ServiçoDeHotel.java) processar a informação disponibilizada. Os dados produzidos na execução da entidade 'ServiçoDeHotel' são retornados ao ControladorDeTarefaSimples (ts_1.java) que corresponde ao passo "ReservaDeHotel" como seus dados de saída (os_1 no ts_1.java).

De novo, usando o ServiçoDeEventos, uma notificação é enviada ao ControladorDeTarefaSimples (implementado pela classe ts_2.java) correspondente ao passo "ReservaDePassagens" anunciando-lhe que sua informação de entrada (oe_1 no ts_2.java da Figura 4.6) está disponível para ser utilizada (corresponde a dep. 2, na Figura 4.6). Esse segundo ControladorDeTarefaSimples (ts_2.java) requer da entidade executora associada com ele (na Figura 4.6 ServiçoDePassagens.java) o processamento da informação a ela disponibilizada. A informação produzida na execução da entidade 'ServiçoDePassagens' é retornada como a saída (os_1) desse segundo controlador (ts_2.java).

Logo depois, uma notificação é enviada ao ControladorDeTarefaComposta (implementado pela classe tc_1.java) o qual controla a totalidade da aplicação de viagem, anunciando-lhe que sua informação de saída (os_1) está disponível. Em seguida, o processo correspondente à aplicação da viagem termina e o resultado fica disponível para quem requisitou o processo.

4.4 RESULTADOS DA EXECUÇÃO

A Figura 4.7 ilustra a interface de acesso à plataforma. Ela permite ao usuário registrar sua entrada na plataforma, como um usuário conhecido, e selecionar os serviços a serem invocados. A informação capturada do campo 'password' permite à plataforma iniciar os processos internos para realizar a autenticação do usuário e posteriormente verificar as autorizações dadas a ele para usar os serviços a serem por ele selecionados.

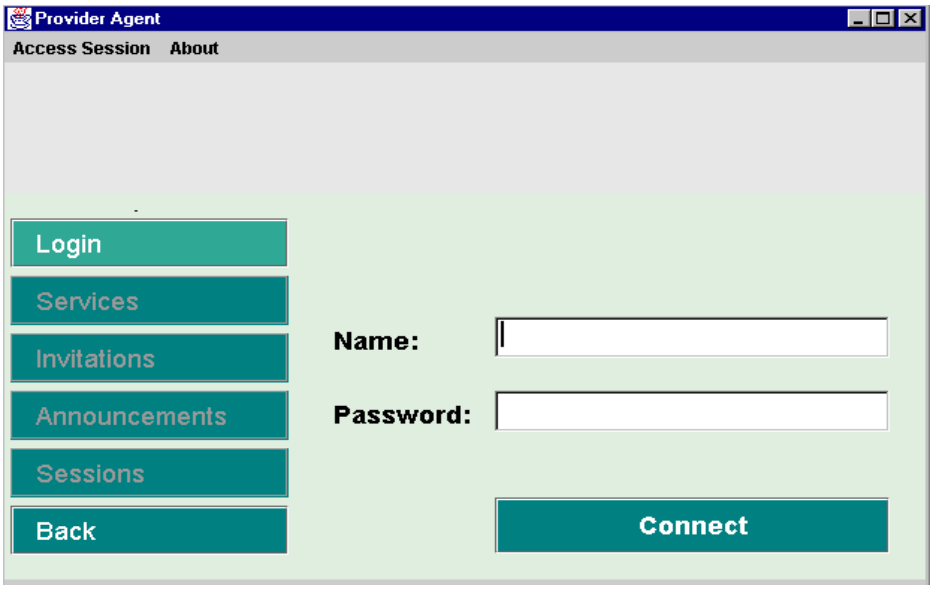


Figura 4.7 Interface de entrada à plataforma

Se o resultado do processo de autenticação e autorização for positivo, o usuário recebe a lista com os serviços disponíveis para ele. A Figura 4.8 mostra uma possível lista de serviços (a lista difere dependendo do usuário e seus direitos de uso). Em particular, nota-se a existência dos serviços "Hotel_Service", "Tickets_Service" e "Workflow_ Engine_ Service". Embora cada um dos serviços "Hotel_Service" e "Tickets_Service" possua uma interface independente para seu uso particular, eles serão combinados, conforme a definição do processo de workflow, representada no Tabela 4.1, pelo serviço "Workflow_Engine_Service".

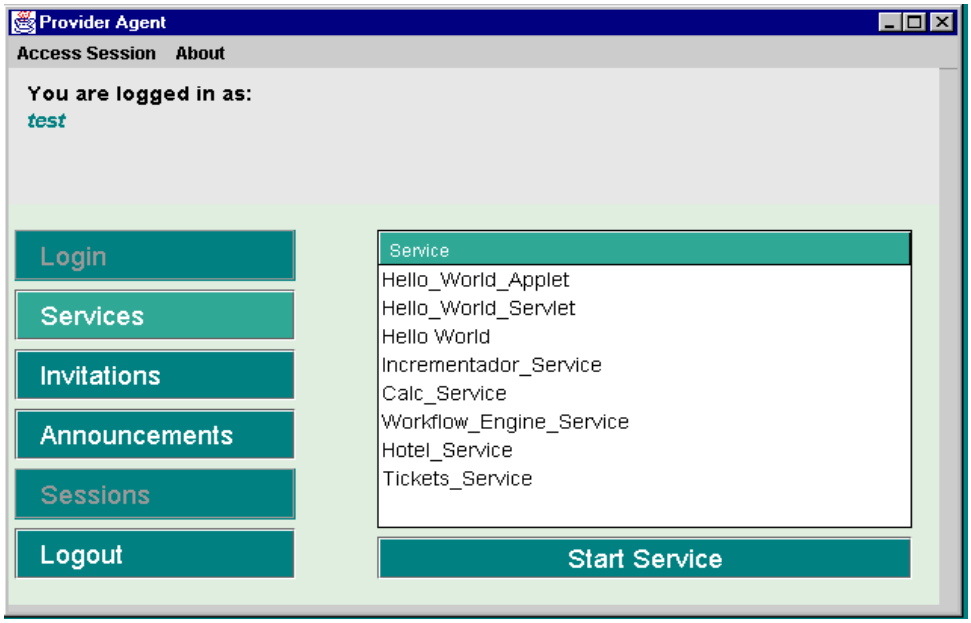


Figura 4.8 Serviços disponíveis para um usuário particular

Uma vez que o usuário tenha selecionado, na interface gráfica mostrada da Figura 4.8, o serviço "Workflow_Engine_Service", ele recebe a interface mostrada na Figura 4.9. Os primeiros quatro campos da interface são usados para capturar a informação que a máquina de workflow requer: (i) os serviços a serem encadeados e (ii) os tipos de dependência entre eles. Os outros campos capturam a informação de contexto que vai ser processada pelos serviços mediados pela plataforma. O último botão permite obter o resultado final do processo

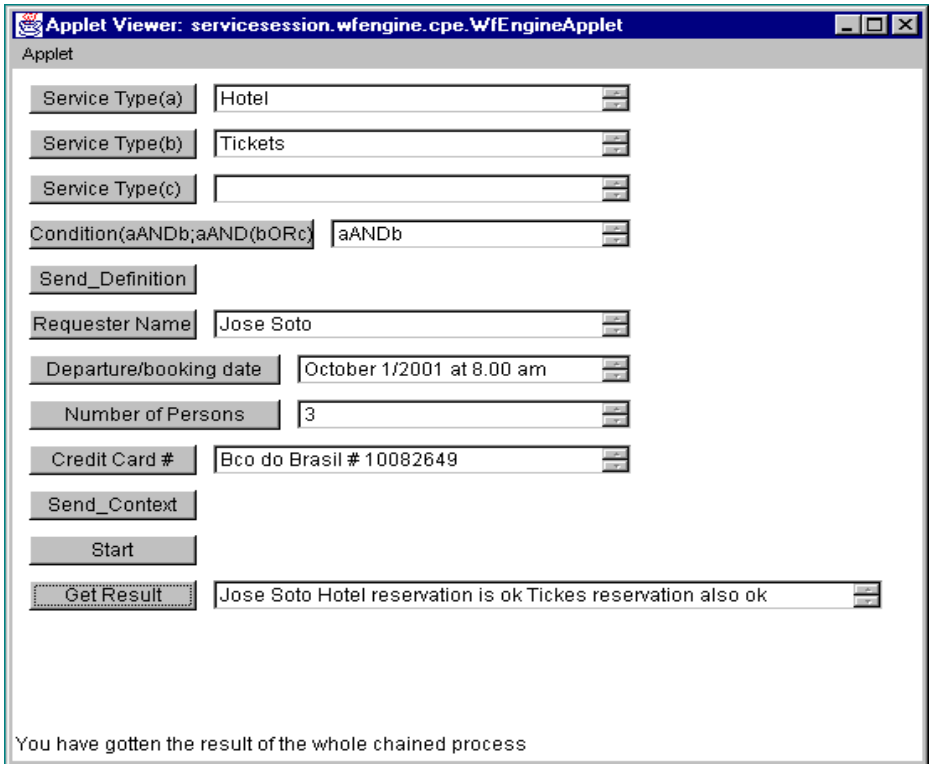


Figura 4.9. Interface do serviço Workflow_Engine_Service

As Figuras seguintes 4.10, 4.11 e 4.12 ilustram o que acontece no monitor da plataforma que oferece e media, para um usuário dela, os serviços: "Hotel_Service", "Tickets_Service" e "Workflow_Engine_Service".

Em particular, a Figura 4.10 ilustra, na janela-monitor da plataforma, alguns registros relacionados com o serviço "Workflow_Engine_Service", o qual é conhecido na plataforma com o nome de "Java Workflow Engine Service". A parte direita da janela registra o recebimento, por parte do serviço da máquina de workflow, da informação associada com os serviços a serem combinados e que foi capturada mediante a interface mostrada na Figura 4.9 anterior. O serviço de workflow recebeu , como primeiro tipo de serviço, "Hotel", como segundo, "Tickets" e como dependência entre eles, a condição "AND", que determina que o serviço "Hotel_Service" e o serviço "Tickets_Service" devem ambos ser realizados para que o processo, como um todo, seja completado. E, de conformidade com a definição de processo, o serviço "Hotel_Service" deve ser realizado primeiro e só depois dele concluir será iniciado o segundo serviço "Tickets_Service".

Embora o processo de workflow ilustrado corresponde ao caso mais simples possível (uma dependência do tipo AND entre só dois serviços) a máquina de workflow implementada é capaz de coordenar processos bem mais complexos compostos por combinações lógicas AND e OR (seqüenciais e paralelas) com qualquer numero de passos. As probas correspondentes foram realizadas fora do ambiente da Plataforma Platin usando o produto Vi-

siBroker 4.0 da Inprise. As mesmas não foram realizadas na Plataforma Platin devido à complexidade da instalação e limitações de tempo.

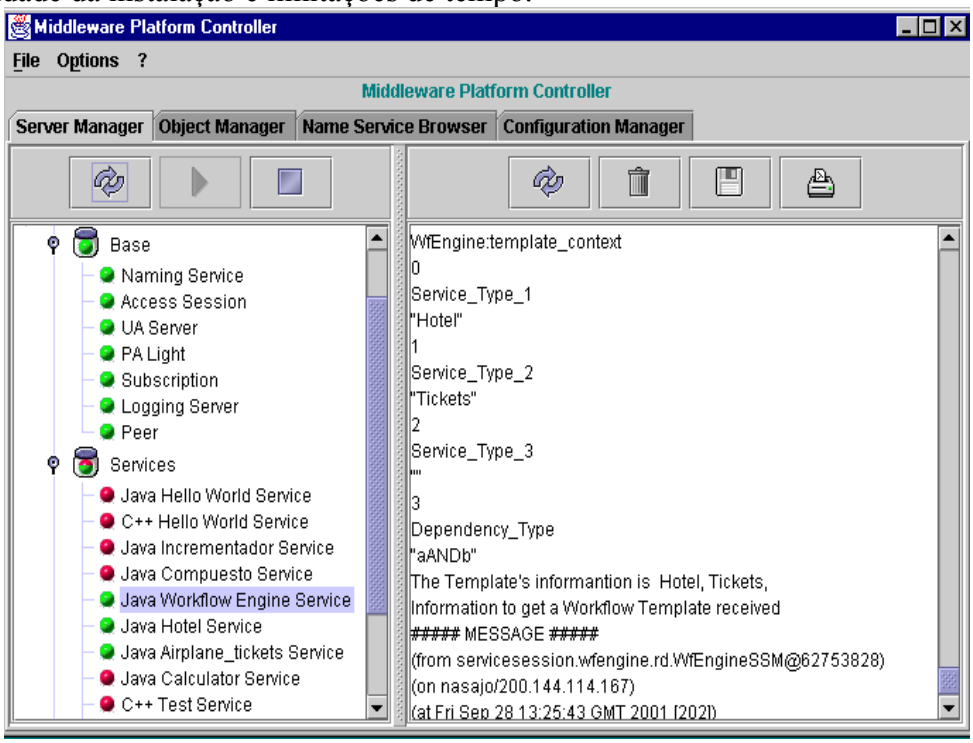


Figura 4.10 Monitor do serviço da máquina de workflow

A Figura 4.11 ilustra, na janela-monitor da plataforma, alguns registros relacionados com o serviço "Hotel_Service", o qual é conhecido na plataforma com o nome de "Java Hotel Service". Especificamente mostra que ele, como primeiro passo do processo, recebe da máquina de workflow a informação (requester_name', 'departure_date', 'number_of_persons' e 'Credit_card_number') dada a ela pelo usuário através da correspondente interface à máquina de workflow.

Este serviço hipotético "Hotel_Service" faz a reserva no Hotel do número de quartos, na data solicitada e o fornece a nome do requisitante. O resultado anterior fica também registrado na janela-monitor do serviço (Figura 4.11).

Depois de ter-se realizado o serviço "Hotel_Service", a informação com o resultado da reserva do hotel é enviada à máquina de workflow. A máquina de workflow examina, na definição do processo, que o serviço "Tickets_Service" tem uma dependência de dados com o serviço "Hotel_Service": especificamente a informação de entrada do serviço "Tickets_Service" é a informação de saída do serviço "Hotel_Service". Assim, a máquina de workflow passa a informação com o resultado da realização do serviço "Hotel_Service" ao seguinte passo no processo, o serviço "Tickets_Service". O anterior fica registrado na janela-monitora da plataforma, correspondente ao serviço "Tickets_Service", conhecido na plataforma com o nome "Java Airplane_tickets Service" (ver Figura 4.12).

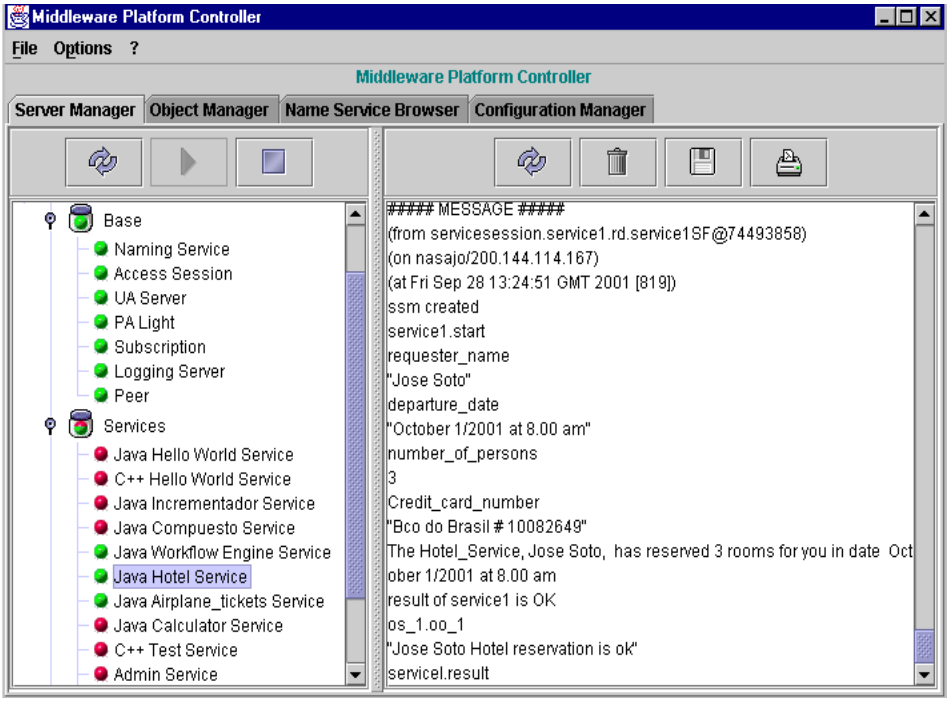


Figura 4.11 Monitor do serviço de Hotel

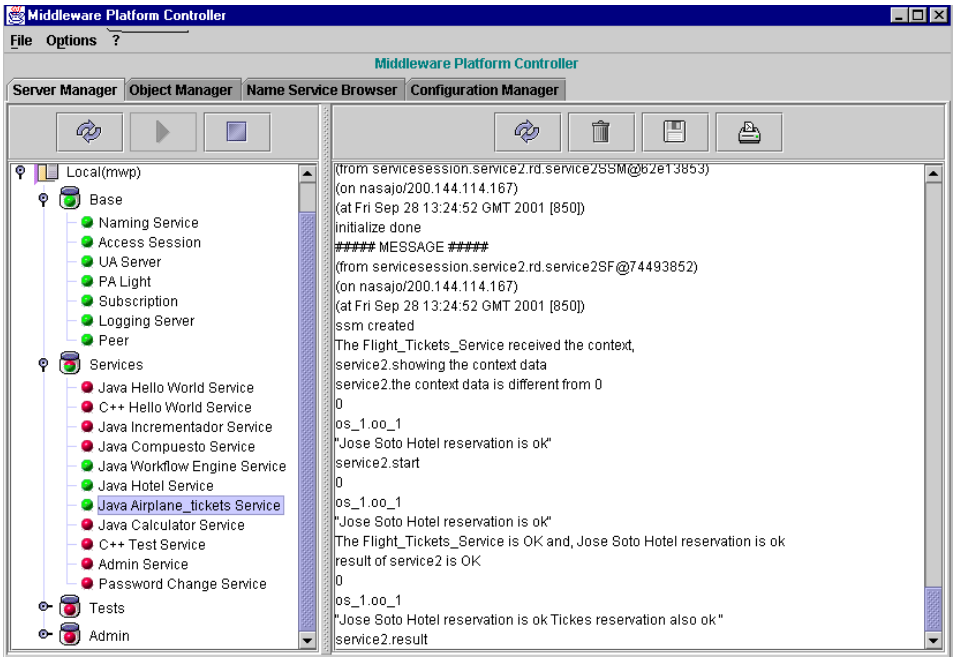


Figura 4.12 Monitor do serviço de passagens aéreas

O serviço “Tickets_Service” depois de ter recebido da máquina de workflow a informação com a confirmação do hotel, como resultado do passo prévio, faz a reserva das passagens aéreas na data solicitada, com o nome do mesmo requisitante. O resultado anterior fica registrado na janela-monitor do serviço (Figura 4.12). A informação, produzida e consolidada neste segundo serviço, é enviada por ele à máquina de workflow. Esse resultado final do

processo pode ser acessado na máquina de workflow (o serviço "Workflow_Engine_Service") pelo usuário que requisitou o processo, através da respectiva interface (Figura 4.9).

Independentemente, cada um dos serviços de hotel e passagens aéreas podem ser invocados fora do contexto de um processo de workflow, usando suas respectivas interfaces. As Figuras seguintes, 4.13 e 4.14, ilustram a interface de acesso ao serviço "Java Hotel Service" e os resultados dele na janela-monitor da plataforma, fora do contexto de um processo de workflow.

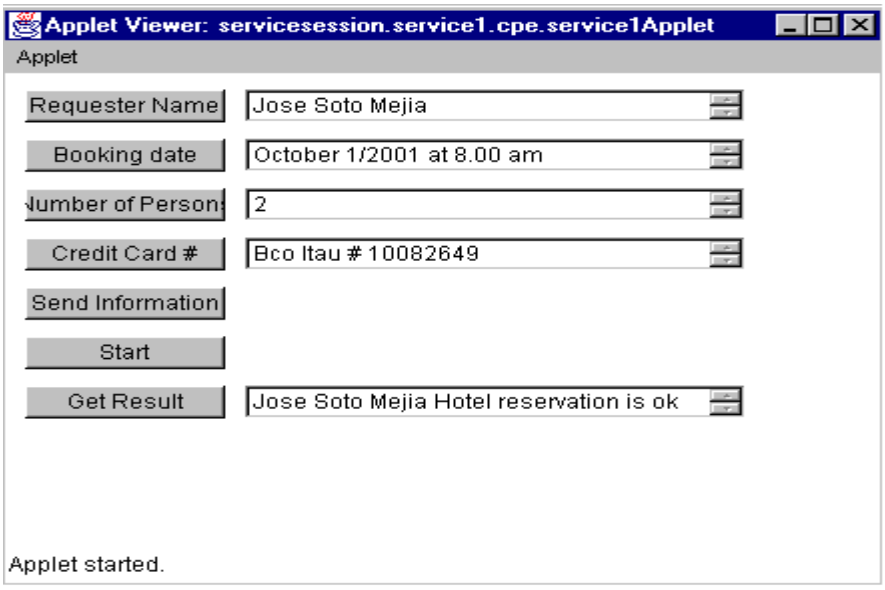


Figura 4.13 Interface ao serviço de Hotel

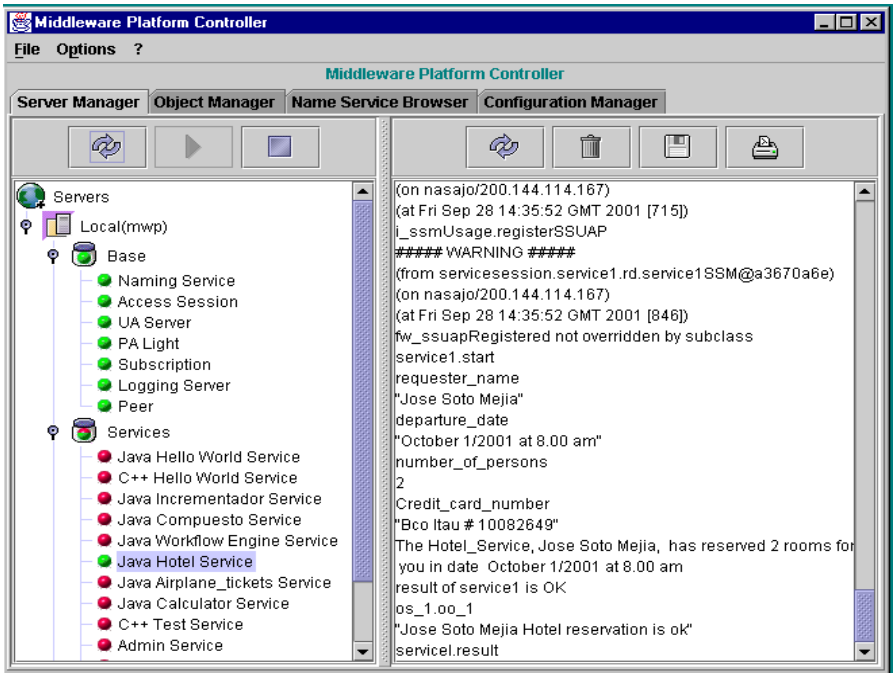


Figura 4.14 Monitor do serviço de Hotel fora do contexto do workflow

4.5 INTERAÇÕES BÁSICAS NA PLATAFORMA

Na Figura 4.15 são ilustradas algumas das interações básicas entre os componentes que fazem parte do domínio do usuário (lado do cliente) da plataforma Platin e vários dos componentes que compõem a plataforma, incluindo os serviços por ela mediados: Hotel_Service, Tickets_Service e Workflow_Engine_Service (a máquina de workflow).

Por simplicidade, vamos assumir que o usuário já tenha entrado na plataforma, tenha sidautenticado e autorizado por ela e como conseqüência recebido a lista com os serviços para ele disponíveis. (esta situação corresponde ao passo ilustrado na Figura 4.8).

Na seqüência são descritas as interações ilustradas na Figura 4.15.

1 O usuário seleciona um dos serviços disponíveis (por exemplo o serviço 'Workflow_Engine_Service').

2 No lado do cliente, um agente de software representante do domínio servidor (PA-Provider Agent, na Figura 4.14) lê a informação associada com o serviço requisitado e instrui o 'browser' para que carregue a correspondente URL (uma referência a um CGI script).

3 O 'browser' envia uma requisição HTTP GET ao servidor de Web.

4 O servidor invoca o CGI script referenciado e constrói uma página HTML que contem a referência a um applet (ssUAP) correspondente à interface gráfica do serviço selecionado (a interface da máquina de workflow).

5 A página HTML é retornada ao 'browser'.

6 O 'browser' encontra o 'tag' do 'applet' e faz o 'download' do seu código

7 O 'browser' instancia o 'applet'.

8,9,10,11 O 'applet' (ss-UAP) interage com o agente provedor do serviço (PA) para que este último (PA) interaja com o agente representante do cliente (UA- User Agent na plataforma) no domínio da plataforma, para que inicie o serviço.

12 O agente do usuário (UA), no domínio do servidor (a plataforma), interage com o componente da plataforma 'Subscrição' (Subscription) o qual contém as referências às fabricas de todos os serviços (SF- Service Factory) mediados pela plataforma.

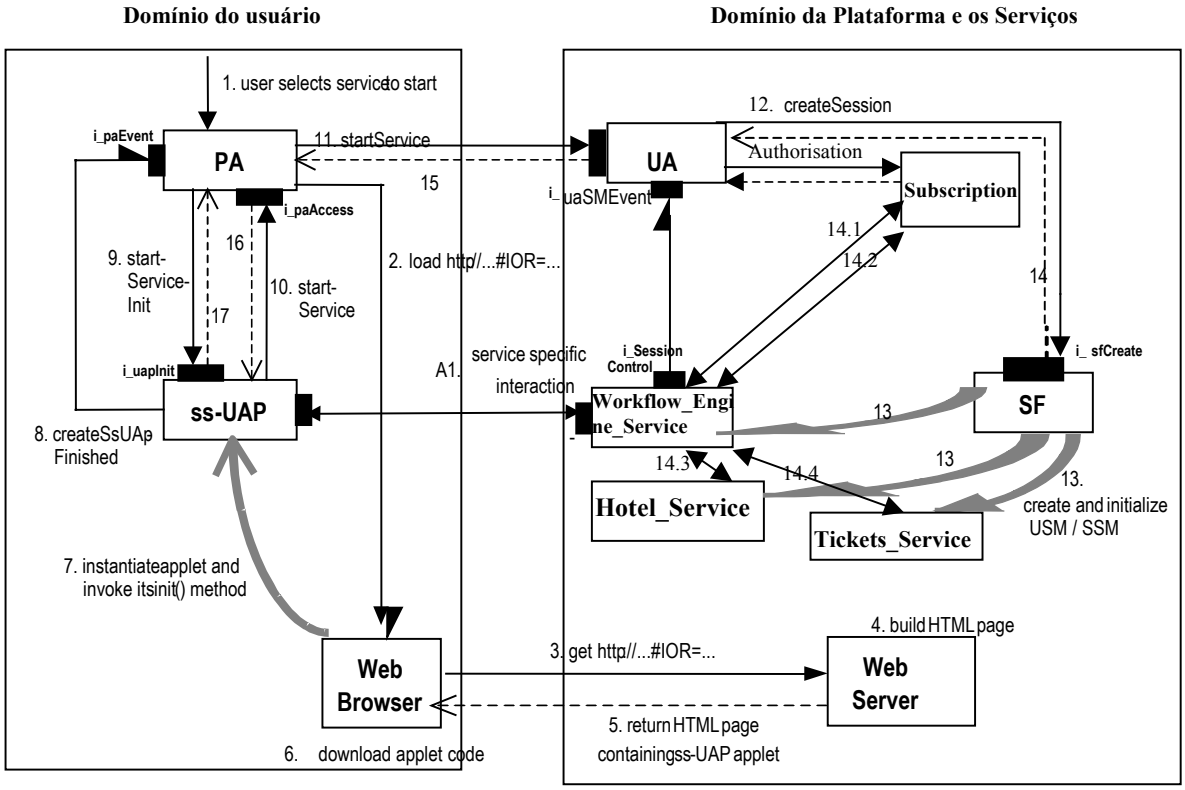


Figura 4.15 Interações entre alguns dos componentes da Plataforma Platin, a máquina de workflow e os serviços Hotel_Service e Tickets_Service.

13 A fábrica de serviços (SF) correspondente instancia a máquina de workflow.

14 A fábrica de serviços (SF) retorna ao agente representante do usuário (UA), no domínio da plataforma, informação (referências a várias interfaces) para que ele possa controlar o serviço recentemente instanciado pela fábrica (SF).

14.1, 14.2 A máquina de workflow, uma vez ativa, ao interpretar a definição do processo, fica ciente da necessidade de invocar os serviços "Hotel_Service" e "Ticket_Service" e procura conhecer as referências particulares de cada um deles no componente de 'Subscrição' (Subscription) da plataforma.

14.3, 14.4 A máquina de workflow, já conhecendo as referências a cada um dos serviços a ser combinados, invoca cada um deles de acordo com a seqüência especificada na definição do processo.

15 O agente representante do usuário (UA) no domínio da plataforma retorna a informação sobre a sessão do serviço ao representante do servidor (PA) no lado do cliente (domínio do usuário).

16 No lado do cliente (domínio do usuário), o agente representante do lado servidor (PA) entrega a informação sobre a sessão do serviço à interface do serviço (ss-UAP) inicialmente invocado pelo cliente. O cliente fica ao comando da interface ao serviço.

A seguinte Figura 4.16 ilustra algumas das interações entre a máquina de workflow, os serviços (‘Hotel_Service’ e ‘Ticket_Service’) que ela coordena e a plataforma Platin.

A plataforma provê um ‘framework’ para desenvolver e integrar serviços nela. Esse ‘framework’ é uma interface para o desenvolvedor da aplicação (Application Programmers Interface API) especificada usando CORBA IDL. A comunicação entre os componentes implementados e os componentes que fazem parte da plataforma é oferecida pelo ‘framework’. Fazendo uso desse ‘framework’, a máquina de workflow e os serviços ‘Hotel_Service’ e ‘Ticket_Service’ implementados em Java foram integrados na plataforma. O ‘framework’ prepara (fw_setup()), conecta (fw_connect()) e registra (fw_registerSessionInterface()), cada um dos serviços dentro da plataforma (ver Figura 4.16).

A máquina de workflow, como um serviço da plataforma iniciado pelo cliente, lê (read_workfow_schema()) e interpreta a definição de workflow da Tabela 4.1. De conformidade com essa definição de processo e dentro da mesma sessão de serviço a máquina de workflow inicia sessões de acesso (fw_startServiceWithinAccessSession()) a cada um dos serviços a serem combinados, para o que busca (searchInterfaceList()) e recupera as referências a cada um deles, fazendo uso do ‘framework’.

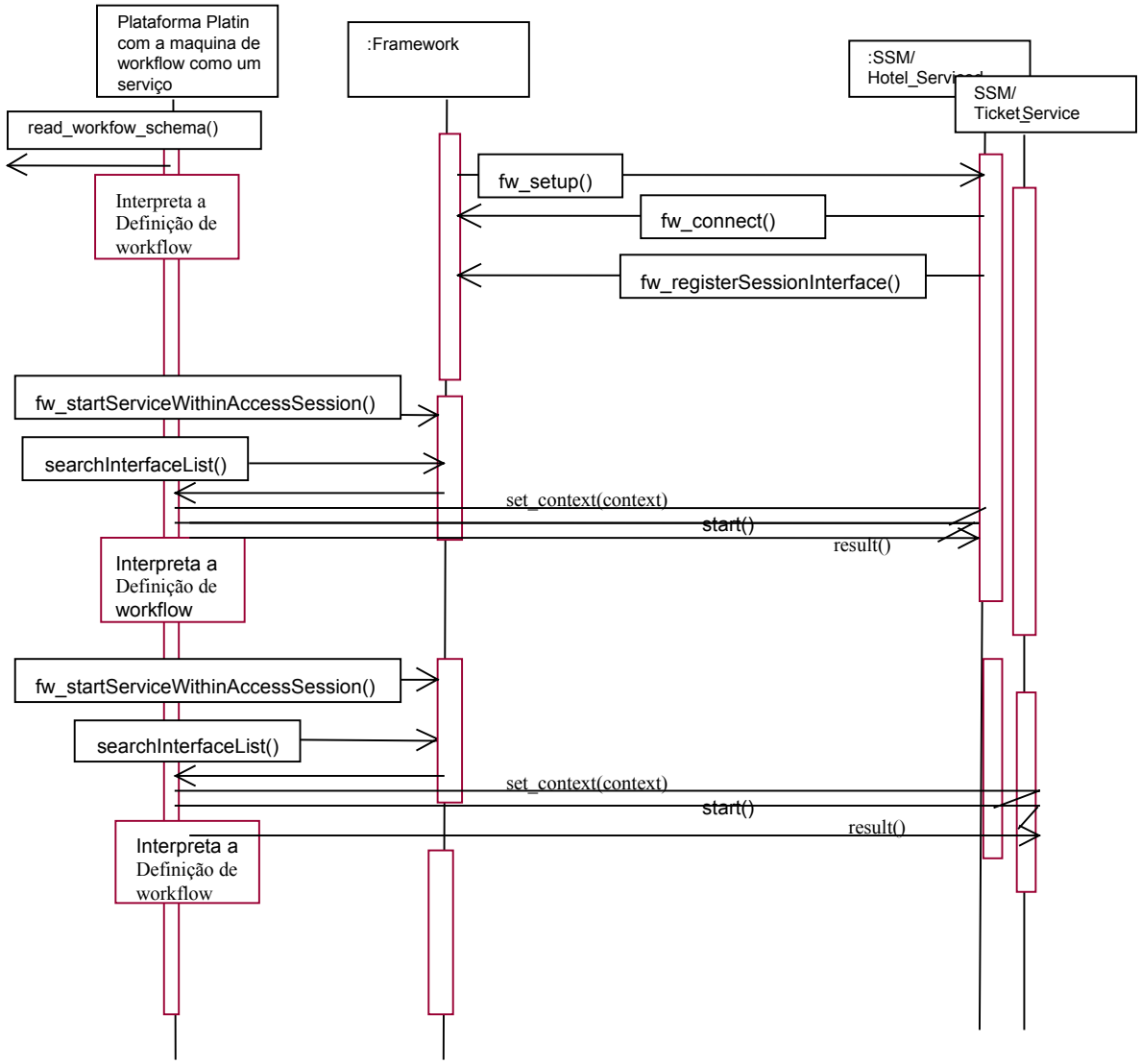


Figura 4.16 Interações entre a máquina de workflow, o 'framework' e os serviços 'Hotel_Service' e 'Ticket_Service'.

O seguinte trecho em Java ilustra como o serviço da máquina de workflow inicia, dentro de sua própria sessão de serviço (fw_startServiceWithinAccessSession()), uma nova sessão para acessar o serviço 'Hotel_Service'. Logo depois busca na lista de interfaces (searchInterfaceList()) da plataforma a referência a esse serviço, para o qual a referência ao objeto CORBA obtida é delimitada ('narrow') ao tipo (i_service1) correspondente ao tipo do serviço Hotel_Service.

```
fw_startServiceWithinAccessSession("Hotel_Service", ssProps, sessionInfoH);
org.omg.CORBA.Object cmpInterface1 = Util.searchInterfaceList(sessionInfoH.value.itfs, "service1::i_service1");
idl.service1.i_service1 service1Itf = i_service1Helper.narrow(cmpInterface1);
```

A referência 'service1Itf' ao serviço 'Hotel_Service' (e de maneira análogo para o serviço 'Ticket_Service') é usada pela máquina de workflow, durante a execução, no momento certo, de acordo com a definição do processo. A máquina verifica no ControladorDeTarefaSimples apropriado, que tem associado com cada serviço, a satisfação de sua correspondente 'Entrada' e logo depois inicia sua entidade executora (no caso o Serviço de Hotel, que está associado com a referência 'service1Itf', fazendo uso de uma Thread (para possibilitar certo tipo de comportamento assíncrono quando for invocar paralelamente a execução de vários serviços).

4.6 INSTALAÇÃO DA PLATAFORMA E CARACTERÍSTICAS DO SISTEMA

A configuração da mini rede com três nós, ilustrada na Figura 4.17, serviu como sistema de teste na integração da máquina de workflow na Plataforma. Podem distinguir se três domínios: (i) o domínio do cliente, (ii) o domínio da Plataforma com a máquina de workflow integrada nela, e (iii) o domínio dos provedores dos serviços (Hotel_Service e Tickets_Service).

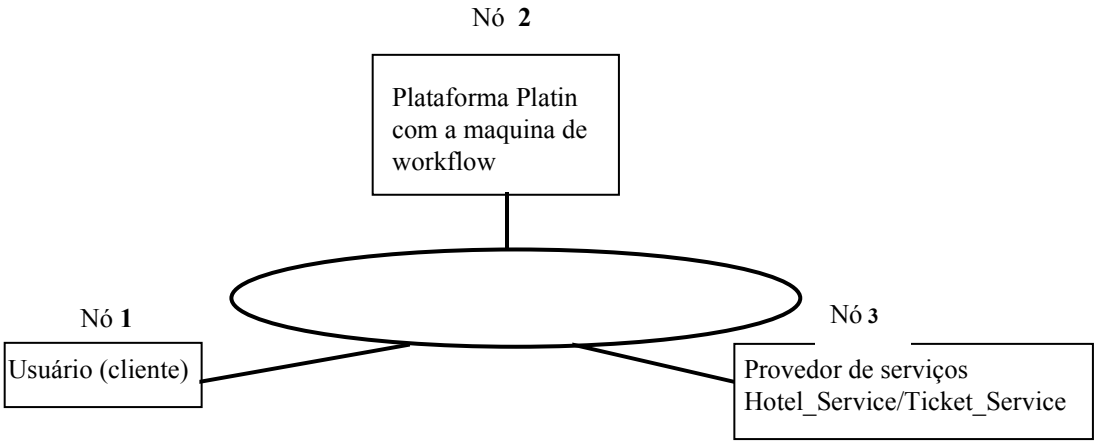


Figura 4.17. Configuração da instalação do cliente, a Plataforma e os serviços

A seguir estão listadas as características do sistema e do software usado na instalação da Plataforma sob o ambiente Windows.

IBM PC AT Compatible
Windows NT 4.0 + Service Pack 4
Microsoft TCP/IP for Windows NT
x86 Family6, 501 Mhz (PII 300 ou compatível é suficiente)
17 752 MB hard disk (65 MB disponível é suficiente)
SVGA com 4MB de memória (VGA é suficiente, 256 cores são requeridos para os Java Applets)
522 MB de RAM (128 MB são suficientes)
Apache 1.3 server (suporta Perl para Win 32)

Java 2 Runtime Environment, Version 1.2.2

Os computadores estavam integrados por uma LAN.

O seguinte software comercial foi usado para suportar a instalação:

- Visibroker 3.3 C++ for Windows
- Visibroker 3.4 Java for Windows
- ObjectStore 6.0 Service Pack 4

4.6 COMENTÁRIOS SOBRE A IMPLEMENTAÇÃO

O metamodelo unificado visa definir uma semântica comum que permita modelar tanto as definições de processo de workflow quanto os sistemas de software que dão suporte ao processo de negócio. Isto implica, usando a semântica do metamodelo unificado, poder especificar a lógica da definição de um processo de workflow e ao mesmo tempo poder especificar o desenho do sistema de software (a aplicação baseada em workflow) que daria suporte ao processo de negócio que está sendo representado mediante a definição do processo de workflow. No caso particular, apresentado neste capítulo, isto corresponde a especificar a lógica da definição do processo mostrada na Figura 4.3 e a especificar o desenho do sistema de software que suporta a aplicação de viagens (Figura 4.4).

O trabalho de implementação apresentado neste capítulo está centrado na implementação de uma máquina de workflow que pode especificamente interpretar um subconjunto da linguagem textual baseada na semântica do metamodelo unificado proposto. Este enfoque permite, uma vez padronizado um metamodelo de definição de processos de workflow e sua representação textual, tornar a implementação das máquinas de workflow independente das definições de processo de workflow a serem executadas. Neste caso, cada máquina teria que ter um componente -interpretador da notação textual padronizada (correspondente à semântica do metamodelo padronizado) que a mapeie ao ambiente de execução específico da máquina de workflow destino.

O segundo aspecto da implementação, que está relacionado com a utilização da máquina de workflow, dentro do contexto de uma aplicação baseada em workflow, tem mostrado: (i) que, em princípio, a inclusão de funcionalidades de workflow dentro da plataforma Platin é uma forma plausível de combinar e coordenar serviços e (ii) que a inclusão deste tipo de funcionalidade pode ser alcançado respeitando a perspectiva operacional da plataforma, como a abertura de uma sessão de serviço particular para acessar um determinado serviço e a autenticação e autorização do usuário do serviço para poder usá-lo.

4.8 REFERÊNCIAS DO CAPÍTULO

[4.1] Timmers Paul, "Business Models for Electronic Markets", http://europa.eu.int/ISPO/ecommerce/publications.html

[4.2] Eckert K, Körner E, Schoo P. "Introduction to Middleware Platform", PLATIN Platform Documentation version 1.0, 2001. FOKUS-Research Institute for Open Communication System, Berlin.

[4.3] Giordani, Benito. "Protótipo de um Sistema de Gerenciamento de Workflows Baseado em Eventos", Departamento de Engenharia de Computação e Automação Industrial. Dissertação de Mestrado, Universidade Estadual de Campinas –UNICAMP-, Brasil, Maio de 1999.

[4.4] "CORBA 2.2 Specification- Complete formal CORBA/IIOP 2.2 Specification" Object Management Group, Document Number: formal/98-07-01; 1998. [ftp://ftp.omg.org/pub/docs/formal/98-07-01.pdf]

[4.5] Schenk, M. "OMG White Paper Open Service Marketplace", OMG Document, May 2000.[ftp://ftp.omg.org/pub/docs/telecom/00-05-02.pdf]; 2000.

[4.6] TINA-C. Service Architecture 5.0, June 1997, http://www.tinac.com/specifications/specifications.htm

CAPÍTULO 5

CONCLUSÕES E TRABALHOS FUTUROS

5.1 CONCLUSÕES FINAIS

O atual perfil EDOC suporta os requerimentos para especificar o projeto orientado a objetos de um sistema de computação empresarial (sistema de software que provê suporte para a realização de um conjunto integrado de processos de negócio numa empresa) num ambiente de computação distribuída e usando um modelo de componentes. A correta implementação desses sistemas requer que a operação do sistema esteja diretamente relacionada com os processos de negócio que ela suporta. A tecnologia de sistemas de workflow é atualmente reconhecida como essencial às organizações que necessitam automatizar seus processos de negócio. A especificação daquelas partes do processo de negócio a serem automatizadas é realizada usando os elementos de modelagem de um dado metamodelo para definição de processos de workflow.

Este trabalho mostra:

1) Que é possível configurar um metamodelo unificado que dê suporte integrado:

- à definição de processos de workflow associados com processos de negócio e
- ao projeto do sistema de software que provê o suporte para a automação dos processos de negócio.

O enfoque apresentado na tese, ao propor um metamodelo unificado que suporte tanto a definição de processos de workflow quanto o projeto do sistema de software que suporta a automação do processo de negócios associado, é um passo a mais no sentido de diminuir os problemas de desentendimento entre os modeladores da lógica do processo de negócio (workflow) e os desenvolvedores do software de suporte. Trata-se de suporte importante (semântico e sintático) para a produção de software de melhor qualidade, no sentido de que a operação dela esteja diretamente relacionada com o processo que pretende suportar, além de prover os fundamentos para minimizar problemas de interoperabilidade entre diferentes produtos implementados por fornecedores independentes.

2) Que é possível configurar um metamodelo unificado que suporte a especificação tanto do controle baseado em dados quanto do controle baseado em eventos.

Dado que os processos de negócio podem ser baseados em eventos, a automação deles exige um correspondente metamodelo de workflow que suporte a especificação dos conceitos de eventos no contexto da definição de um processo de workflow. O metamodelo unifi-

cado proposto dá suporte à especificação tanto do controle baseado em dados quanto do controle baseado em eventos.

3) Que é possível introduzir uma notação textual correspondente ao metamodelo unificado.

Apoiada nos metaconceitos introduzidos no metamodelo unificado, o trabalho apresenta um esboço de uma linguagem textual genérica (no sentido de ser independente da plataforma), para representar a lógica dos processos de workflow. Esta linguagem é preprocessada pela máquina de workflow implementada, que a interpreta para depois executá-la. Embora a representação textual proposta na tese, não seja uma linguagem padrão, é genérica no sentido de permitir uma modelagem do processo de workflow independente da plataforma na qual vai ser executado e está, nesse sentido, no caminho da atual tendência que busca padronizar um metamodelo padrão de workflow. O enfoque apresentado também é consistente com o da arquitetura orientada por modelos MDA (Model Driven Architecture) que enfatiza a formulação de modelos independentes da plataforma PIM (Plataform Independent Model) e posterior mapeamento deles às várias tecnologias de implementação (ver seção 1.2.3).

No contexto deste enfoque, de formular uma linguagem padrão independente da plataforma, cada fornecedor de sistemas de gerenciamento de workflow só teria que prover um preprocessador que interprete a lógica do processo de workflow e alimente sua máquina de workflow proprietária. Este enfoque oferece aos desenvolvedores liberdade na implementação de suas máquinas de workflow, sem requerer deles a padronização das mesmas. Facilita, ao mesmo tempo, a reutilização de máquinas de workflow legadas.

4) Que é possível interpretar e executar uma definição de processo de workflow em termos da notação textual proposta, através da implementação de uma máquina de workflow.

A integração e uso da máquina de workflow na Plataforma Platin permite deduzir que ela é um mecanismo plausível para combinar e coordenar serviços, inicialmente integrados como independentes, dentro dessa plataforma.

5.2 TRABALHOS FUTUROS

A seguir descrevem-se alguns desenvolvimentos coerentes com o tema desta dissertação e que podem ser realizados em trabalhos futuros.

1) A abordagem proposta neste trabalho e formalizada com o nome de metamodelo unificado busca minimizar os problemas de desentendimento entre os modeladores da lógica do processo de workflow e os desenvolvedores do seu software de suporte. O modelo do processo de workflow, obtido graficamente, usando os metaconceitos que fazem parte do metamodelo unificado, deve ser mapeado à notação textual proposta para que possa ser reconhecido, interpretado e executada pela máquina de workflow. Considera-se como trabalho futuro a implementação de um mecanismo apropriado que automaticamente realize esse mapeamento.

2) No caminho para uma abordagem mais genérica, com respeito à generalidade da notação textual proposta, pode ter sentido explorar alternativas para sua formalização, em termos de uma ontologia para a definição de processos de workflow, consistente com o metamodelo unificado proposto. E, seguindo as idéias da proposta de 'Linguagem de Definição de Processos em XML da WfMC' [5.1], que mapeia ao XML[5.2] a 'Linguagem de Definição de Processos de Workflow da WfMC' [5.3] (The WfMC Workflow Process Definition Language), correspondente à especificação da interface 1 [5.4] (ver Figura 1.1) , propor, por exemplo, um DTD[34] adequado (Document Type Definition), específico para workflow.

3) A implementação de uma máquina de workflow, capaz de executar definições de processo de workflow codificadas em XML, de conformidade com um DTD específico para workflow, é também de interesse, visando-se o reuso de diferentes definições de processo.

4) Outra linha de desenvolvimento deste trabalho e que é de interesse futuro explorar é a pertinência e possibilidade de adequar o uso de uma máquina de workflow na Internet disponibilizando- a como um Serviço Web.

Os chamados Serviços Web (Web Services) [35] [5.5] tornam possível a interoperabilidade via padrões abertos como XML[36] e HTTP [5.6]. Os sistemas pretendem usar SOAP (Simple Object Access Protocol) [5.7] para rodar as aplicação do tipo serviço Web. Dentro deste contexto dos serviços Web seria de interesse implementar uma máquina de workflow compatível com o protocolo SWAP (Simple Workflow Access Protocol) [5.8], [5.9]. A idéia básica de SWAP centra-se em definir um enlace entre o modelo de interfaces da Facilidade de Workflow da OMG (Workflow Management Facility) [5.10] e um protocolo de interação baseado em HTTP, e não no protocolo IIOP de CORBA.

A definição de um protocolo específico para workflow está fundamentada nos requerimentos únicos envolvidos num serviço de workflow: o serviço, uma vez invocado, pode tomar desde minutos até horas (anos) para ser completado. Durante esses período de tempo podem ocorrer mudanças e não é evidente que os atuais protocolos de notificação e de transação possam ser escalados para incluir este tipo de assincronismo.

---

[34] Um DTD (Document Type Definition) pode acompanhar um documento XML, essencialmente definindo formalmente as regras do documento como, por exemplo, que nomes podem ser usados como elementos, onde podem estar localizados e as relações estruturais entre os elementos que fazem parte do arquivo XML

[35] Os Serviços Web (Web Services) constam de um conjunto de tecnologias projetadas para facilitar o oferecimento de serviços pelas intranets e Internet. Os serviços Web, em essência, integram PCs numa plataforma computacional, na qual os usuários podem trabalhar com os serviços via browsers Os serviços propriamente ditos rodam sobre servidores Web (Web-based servers) e não nos PCs, movendo desta maneira as funcionalidades do 'desktop' para a Internet. O provedor de serviços Web registra e lista seus serviços na Internet num Diretório de Serviços Web, utilizando um registro baseado no protocolo UDDI (Universal Description Discovery and Integration protocol) o qual permite ao serviço Web ser encontrado na Internet.

[36] XML é o padrão mais importante nos serviços Web. Os 'tags' do XML provêem informação acerca dos dados num documento em quase qualquer plataforma, permitindo a comunicação entre diferentes plataformas.

5.3 REFERÊNCIAS DO CAPÍTULO

[5.1] Workflow Management Coalition. "Workflow Standard -Interoperability Wf-XML Binding", WfMC-TC-1023, Version 1.0 May 2000.
http://www.wfmc.org/standards/docs/Wf-XML-11.pdf

[5.2] XML. Extensible Markup Language
http://www.microsoft.com/xml
XML. Extensible Markup Language. W3C Recomendation
http://www.w3.org/TR/REC-xml-19980210

[5.3] Workflow Process Definition Language- XML Process Definition Language
Document Number WFMC-TC-1025-Draft 0.03, May 22, 2001
http://www.wfmc.org/standards/docs/xpdl_010522..pdf

[5.4] Interface 1 – Process Definition Interchange- Process Model, v 1.1WfMC-TC-1016-P, Workflow Management Coalition, October 29, 1999;
http://www.wfmc.org/standards/docs/TC-1016-P_v11_IF1_Process_definition_Interchange.pdf

[5.5] Steven J. Vaughan-Nichols. "Web Services: Beyond the Hype". Computer. IEEE, February 2002, Volume 35, Number 2, pag 18-21.

[5.6] H.J Nielsen. "Hypertext Transfer Protocol" Overview page, August 1998,
http://www.w3.org/MarkUp/

[5.7] A. Skonnard, "The Simple Object Access Protocol", Jan 2000.
http://www.msdn.microsoft.com/library/periodic/period00/soap.htm
http://www.w3.org/TR/2000/NOTE-SOAP-20000508

[5.8] K. Swenson, "Simple Workflow Access Protocol (SWAP)," Internet draft, 7 Aug. 1998.
http://www.ics.uci.edu/~irus/wisen/wisen98/presentations/Swenson/

[5.9] Workflow Management Coalition. "Workflow Standard -Interoperability Wf-XML Binding", WfMC-TC-1023, Version 1.0 May 2000.
http://www.wfmc.org/standards/docs/Wf-XML-11.pdf

[5.10] Juergen Boldt. Workflow Management Facility Specification (WMF), v1.2, OMG Document Number: formal/00-05-02; 2000.
[http://ftp.omg.org/pub/docs/formal/00-05-02.pdf]

CAPÍTULO 6

REFERÊNCIAS BIBLIOGRÁFICAS

REFERÊNCIAS DO CAPÍTULO 1

[1.1] Knutilla, A.; Schlenoff, C.; Ray, S.; Polyak, S. T.; Tate, A.; Cheah S.C.; Anderson, R. C. "Process Specification Language: An Analysis of Existing Representations". [http://www.mel.nist.gov/psl], último acesso (10/10/2001).

[1.2] The Workflow Management Coalition, WfMC.[http://www.wfmc.org]

[1.3] Workflow Management Coalition. Terminology and Glossary. Document Number WFMC-TC-1011, June 1996.

[1.4] Workflow Reference Model. Document Number WfMC-TC-1003 v1.1 (Jan 95). [http://www.wfmc.org]

[1.5] Interface 1 – Process Definition Interchange- Process Model, v 1.1WfMC-TC-1016-P, Workflow Management Coalition, October 29, 1999; [http://www.wfmc.org]

[1.6] Workflow Client API Specifications (WAPI); WfMC-TC-1002, v2.0 (Jul 98). [http://www.wfmc.org]

[1.7] Workflow Interoperability- Abstract Specifications; WfMC-TC-1012, v2.0 (Dec 99). [http://www.wfmc.org]

[1.8] Workflow Audit Data Specification; WfMC-TC-1015, v1.1 (Sep 98). .[http://www.wfmc.org]

[1.9] W. M. P van der Aalst, K. M. van Hee and G. J. Houben. "Modeling and Analysing Workflow using a Petri-net based Approach". Proc. of the 2nd Workshop on Computer Supported Cooperative Work, Petri Nets and Related Formalisms. [http://wwwis.win.tue.nl/~wsinwa/wfm-adv.ps]

[1.10] 'The Object Management Group (OMG)' , http://www.omg.org.

[1.11] Juergen Boldt. Workflow Management Facility Specification (WMF), v1.2, OMG Document Number: formal/00-05-02; 2000. [http://ftp.omg.org/pub/docs/formal/00-05-02.pdf]

[1.12] Richard Soley (ed.). "Object Management Architecture Guide", Third Edition, Wiley, June 1995. OMG Document ab/97-05-5; 1997 [http://ftp.omg.org/pub/docs/ab/97-05-5.pdf]

[1.13] Ms. Linda Heaton. “Unified Modeling Language Specification v1.4 “; 2001; OMG Document Number: formal/01-09-67a formal/01-09-80 [[http://ftp.omg.org/pub/docs/formal/01-09-67.pdf]

[1.14] “UML Profile for Enterprise Distributed Object Computing”, –Request for Proposal, OMG Document Number: ad/99-03-10; 1999.
[http://ftp.omg.org/pub/docs/ad/99-03-10.pdf]

[1.15] A.P. Sheth, W. van der Aalst, and I.B. Arpinar, “Process Driving the Networked Economy”, IEEE Concurrency, Vol. 7, No. 3, July-September 1999, [http://dlib.computer.org/pd/books/pd1999/pdf/p3018.pdf]

[1.16] Arpinar S; Dogac A; Tatbul N. “An Open Electronic Marketplace through Agent-based Workflows:MOPPET”.
[http;//www.srdc.metu.edutr/srdc_publications.html]

[1.17] A. Dogac, et. al., “A Workflow-based Electronic Marketplace on the Web”, [http://www.acm.org/sigmod/sigmod_record/issues/9812/SPECIAL/dogac.pdf.gz]

[1.18] Bussler Christoph. “Enterprise-Wide Workflow Management.”. IEEE Concurrency, Vol. 7, No. 3, July-September 1999.[http://dlib.computer.org/pd/books/pd1999/pdf/p3032.pdf]

[1.19] “UML Extensions for Workflow Process Definition” –Request for Proposal, OMG Document Number: bom/2000-12-11; 2000.
[http://ftp.omg.org/pub/docs/bom/00-12-11.pdf]

[1.20] UML Profile for Enterprise Distributed Object Computing-EDOC Draft Adopted Specification OMG Document Number: ptc/2001-12-04; 2001
[ftp://ftp.omg.org/pub/docs/ptc/01-12-04.pdf]

[1.21] Eckert K, Körner E, Schoo P. “Introduction to Middleware Platform”, PLATIN Platform Documentation version 1.0, 2001. FOKUS-Research Institute for Open Communication System, Berlin.

[1.22] Schenk, M. “OMG White Paper Open Service Marketplace”, OMG Document, May 2000.[ftp://ftp.omg.org/pub/docs/telecom/00-05-02.pdf]

[1.23] Meta Object Facility (MOF) v1.3.1, OMG Document Number: formal/01-11-02; 2001.
[ftp://ftp.omg.org/pub/docs/formal/01-11-02.pdf]

[1.24] “White Paper on the Profile Mechanism”, OMG Document Number: ad/99-04-07; 1999
[ftp://ftp.omg.org/pub/docs/ad/99-04-07.pdf]

[1.25] ISO/IEC & ITU-T: Information technology – Open Distributed Processing – Reference Model - Architecture – ITU-T Recommendation X.903 | ISO/IEC 10746-3

[1.26] ISO/IEC & ITU-T: Information technology – Open Distributed Processing – Reference Model - Enterprise Language – ITU-T Recommendation X.911 | ISO/IEC 15414
OMG Document: ISO-std/01-01-01; 2001
[ftp://ftp.omg.org/pub/docs/ISO-std/01-01-01.pdf]

[1.27] CORBA Component Model, OMG Document Number: ptc/01-11-02; 2001.
[ftp://ftp.omg.org/pub/docs/ ptc/01-11-02.pdf]

[1.28] Enterprise JavaBeans Specification. Sun Microsystems Inc.
[http://www.javasoft.com/products/ejb/newspec.html]

REFERÊNCIAS DO CAPÍTULO 2

[2.1] Interface 1 – Process Definition Interchange- Process Model, v 1.1WfMC-TC-1016-P, Workflow Management Coalition, October 29, 1999; [http://www.wfmc.org]

[2.2] Warne John, "Nortel Revised Submission to the Workflow Management RFP", OMG Document Number: bom/98-03-01, 1998

[2.3] UML Profile for Enterprise Distributed Object Computing-EDOC Draft Adopted Specification OMG Document Number: ptc/2001-12-04; 2001.
[ftp://ftp.omg.org/pub/docs/ptc/01-12-04.pdf]

[2.4] Request for Proposals for a Workflow Management Facility, OMG document Number: cf/97-05-06; 1997.

[2.5] Michael zur Muhlen. "Evaluation of Workflow Management Systems Using Meta Models". Proceedings of the 32nd Hawaii Internacional Conference on System Sciences, 1999, (HICSS-1999)

[2.6] Workflow Process Definition Interface-XML Process Definition Language. Document Number WfMC-TC-1025. Version 0.03 (Draft), May 22, 2001

[2.7] Warne John, "Nortel Revised Submission to the Workflow Management RFP", OMG Document Number: bom/98-03-01, 1998

[2.8] ISSO/IEC&ITU-T: Information Technology – Open Distributed Processing – Enterprise Viewpoint – ITU-T Recomendation X.911-ISSO/IEC 15414.

[2.9] Conrad Bock,Guus Ramackers . Partial Evaluation of the Joint EDOC Submission, August 20, 2001. OMG Document Number: ad/01-08-26
[ftp://ftp.omg.org/pub/docs/ad/01-08-26.pdf]

[2.10] Conrad Bock, Thomas Weigert, Response to Response to the Evaluation of the Joint EDOC Submission, August 27, 2001 . OMG Document Number: ad/01-08-33, [ftp://ftp.omg.org/pub/docs/ad/01-08-33.pdf]

[2.11] "UML Profile for Enterprise Distributed Object Computing", –Request for Proposal, OMG Document Number: ad/99-03-10; 1999.
[http://ftp.omg.org/pub/docs/ad/99-03-10.pdf]

[2.12] Editor, Ms. Linda Heaton. "Unified Modeling Language Specification v1.4 "; 2001; OMG Document Number: formal/01-09-67 a formal/01-09-80[http://ftp.omg.org/pub/docs/formal/01-09-67.pdf]

REFERÊNCIAS DO CAPÍTULO 3

[3.1] UML 1.4 - Chapter 6 - Object Constraint Language Specification
OMG Document Number: formal/01-09-77; 2001.
[http://ftp.omg.org/pub/docs/formal/01-09-77.pdf

[3.2] Juergen Boldt. Workflow Management Facility Specification (WMF), v1.2, OMG Document Number: formal/00-05-02; 2000.
[http://ftp.omg.org/pub/docs/formal/00-05-02.pdf]

[3.3] Editor, Ms. Linda Heaton. "Unified Modeling Language Specification v1.4 "; 2001; OMG Document Number: formal/01-09-67 a formal/01-09-80 [http://ftp.omg.org/pub/docs/formal/01-09-67.pdf]

[3.4] Craig Dewalt, Business Process Modeling with UML. OMG Document:ad/00-02-04; 2000.

[3.5] CIMOSA. A Primer on key concepts, purpose and business values. http://cimosa.cnt.pl/Docs/Primer/primer0.htm , (Accessed : 22/01/20002)]

[3.6] Workflow Management Coalition. Terminology and Glossary. Document Number WfMC-TC-1011, v3.0 (Feb 99).

REFERÊNCIAS DO CAPÍTULO 4

[4.1] Timmers Paul, "Business Models for Electronic Markets", http://europa.eu.int/ISPO/ecommerce/publications.html

[4.2] Eckert K, Körner E, Schoo P. "Introduction to Middleware Platform", PLATIN Platform Documentation version 1.0, 2001. FOKUS-Research Institute for Open Communication System, Berlin.

[4.3] Giordani, Benito. "Protótipo de um Sistema de Gerenciamento de Workflows Baseado em Eventos", Departamento de Engenharia de Computação e Automação Industrial. Dissertação de Mestrado, Universidade Estadual de Campinas –UNICAMP-, Brasil, Maio de 1999.

[4.4] "CORBA 2.2 Specification- Complete formal CORBA/IIOP 2.2 Specification" Object Management Group, Document Number: formal/98-07-01; 1998.
[ftp://ftp.omg.org/pub/docs/formal/98-07-01.pdf]

[4.5] Schenk, M. "OMG White Paper Open Service Marketplace", OMG Document, May 2000.[ftp://ftp.omg.org/pub/docs/telecom/00-05-02.pdf]

[4.6] TINA-C. Service Architecture 5.0, June 1997, http://www.tinac.com/specifications/specifications.htm

REFERÊNCIAS DO CAPÍTULO 5

[5.1] Workflow Management Coalition. "Workflow Standard -Interoperability Wf-XML Binding", WfMC-TC-1023, Version 1.0 May 2000.
http://www.wfmc.org/standards/docs/Wf-XML-11.pdf

[5.2] XML. Extensible Markup Language
http://www.microsoft.com/xml
XML. Extensible Markup Language. W3C Recomendation
http://www.w3.org/TR/REC-xml-19980210

[5.3] Workflow Process Definition Language- XML Process Definition Language
Document Number WFMC-TC-1025-Draft 0.03, May 22, 2001
http://www.wfmc.org/standards/docs/xpdl_010522..pdf

[5.4] Interface 1 – Process Definition Interchange- Process Model, v 1.1WfMC-TC-1016-P, Workflow Management Coalition, October 29, 1999;
http://www.wfmc.org/standards/docs/TC-1016-P_v11_IF1_Process_definition_Interchange.pdf

[5.5] Steven J. Vaughan-Nichols. "Web Services: Beyond the Hype". Computer. IEEE, February 2002, Volume 35, Number 2, pag 18-21.

[5.6] H.J Nielsen. "Hypertext Transfer Protocol" Overview page, August 1998,
http://www.w3.org/MarkUp/

[5.7] A. Skonnard, "The Simple Object Access Protocol", Jan 2000.
http://www.msdn.microsoft.com/library/periodic/period00/soap.htm
http://www.w3.org/TR/2000/NOTE-SOAP-20000508

[5.8] K. Swenson, “Simple Workflow Access Protocol (SWAP),” Internet draft, 7 Aug. 1998.
http://www.ics.uci.edu/~irus/wisen/wisen98/presentations/Swenson/

[5.9] Workflow Management Coalition. “Workflow Standard -Interoperability Wf-XML Binding”, WfMC-TC-1023, Version 1.0 May 2000.
http://www.wfmc.org/standards/docs/Wf-XML-11.pdf

[5.10] Juergen Boldt. Workflow Management Facility Specification (WMF), v1.2, 2000; OMG Document Number: formal/00-05-02
[http://ftp.omg.org/pub/docs/formal/00-05-02.pdf]

Apêndice -Publicações

Durante o desenvolvimento deste trabalho foram realizados os seguintes artigos:

Giordani, B. T.; Soto, M. J.; Mendes, M. J. "Internet Trading of Intangible Goods Based on the Execution of Workflows", *Proceedings of the Thirty-Third Hawaii International Conference on System Sciences (HICSS-33)*, IEEE, January 2000.

Mendes, M. J**;** Soto, M. J; J; Silva, R.C; Carvalho, M; Silva, F.C) o artigo "Integration of Workflow Management Systems with Planning Suites in Supply Chains" aprovado para apresentação em 16th International Conference on CAD/CAM, Robotics and Factories of the Future CARS&FOF 2000 , June 26-28, 2000. Port of Spain, Trinidad W.I. Sponsored by International Society for Productivity Enhancement (ISPE) and The University of the West Indies (UWI)

Silva, R.C ; Soto, M. J; Mendes, M. J; "Integrating a Workflow engine and a MOF repository to an Open service Platform", Apresentado em IFIP PRO-VE'02, *Proc. of the 3rd IFIP Working Conference on Infrastructures for Virtual Enterprises*. Sesimbra, PORTUGAL – 1-3 May 2002

Soto, M. J; Mendes, M. J; "An Unified Approach for Specifying Workflow Process Definitions and the Software Systems that Support them"; SBES'2002, *The 16th Brazilian Symposium on Software Engineering*. Gramado, Rio Grande do Sul, Brazil October 16-18, 2002. (submetido)

Publicação em livro
Collaborative Business Ecosystems and Virtual Enterprises
Edited by Luis M. Camarinha-Matos
Kluwer Academic Publishers, Boston/Dordrecht/London
2002.
Part 6. Workflow Management.
Integrating a Workflow Engine and a MOF Repository to an Open Service Platform.
Cláudio R. M. Silva, José A. Soto, Manuel de Jesus Mendes
pag. 161-168

Lista de Siglas

CGI	Common Gateway Interface
CORBA	Common Object Request Broker Architecture
DPE	Distributed Processing Environment
EDOC	Enterprise Distributed Object Computing
EJB	Enterprise Java Beans
HTML	Hyper Text Markup Language
HTTP	Hyper Text Transport Protocol
IIOP	Internet Inter-ORB Protocol
IOR	Interoperable Object Reference
KTN	Kernel Transport Network
MDA	Model Driven Architecture
MMU	Metamodelo Unificado
ODP	Open Distributed Processing
OMA	Object Management Architecture
OMG	Object Management Group
ORB	Object Request Broker
PA	Provider Agent
PIM	Platform Independent Model
RM-ODP	Reference Model of Open Distributed Processing
SSM	Service Session Manager
SF	Service Factory
ssUAP	Service Specific User Application
TINA	Telecommunications Information Networking Architecture
UA	User Agent
UML	Unified Modeling Language
USM	User Session Manager
WfMC	Workflow Management Coalition