v.8, n.1, p.19-36, abr. 2001

A APLICAÇÃO DA LINGUAGEM DE MODELAGEM UNIFICADA (UML) PARA O SUPORTE AO PROJETO DE SISTEMAS COMPUTACIONAIS DENTRO DE UM MODELO DE REFERÊNCIA

Carlos Alberto Costa
Departamento de Engenharia Mecânica
Universidade de Caxias do Sul (UCS)
Rua Francisco Getúlio Vargas, 1130
95.070-500 – Caxias do Sul – RS
E-mail: cacosta@ucs.tche.br

Resumo

O desenvolvimento de sistemas automatizados de informações, que apóiam as atividades de projeto e manufatura de produtos, deve seguir um modelo como referência para permitir uma melhor compatibilidade e portabilidade de tais sistemas, principalmente quando inseridos num ambiente integrado de engenharia concorrente. Este artigo demonstra como a Linguagem de Modelagem Unificada (UML) pode ser aplicada em conjunto com o Modelo de Referência para Processamento Distribuído Aberto (ISO/RM-ODP), para o apoio ao desenvolvimento de sistemas de informações orientados a objetos. Enquanto o RM-ODP oferece um padrão para representação de diferentes pontos de vistas de tais sistemas, a UML é utilizada como notação para representação de cada uma destas vistas. Um processo baseado em Use Cases é empregado para apoiar a evolução da representação das informações dentro deste modelo de referência. O ambiente de projeto de moldes de injeção é utilizado como exemplo para ilustração dos diagramas da UML.

Palavras-chave: modelagem de informações, RM-ODP, UML.

1. Introdução

O grande avanço dos sistemas computacionais nas áreas de processamento e armazenamento de informações tem permitido a aplicação de tais ferramentas nos mais diversos campos. Tal aplicação tem se destacado sobretudo nas áreas de engenharia (projeto e

fabricação) devido à grande quantidade de informações e decisões envolvidas. Paralelamente a este avanço, filosofias e técnicas como Engenharia Concorrente (CE) tem propiciado formas mais inteligentes e otimizadas de se lidar com as informações envolvidas no desenvolvimento de novos produtos.

Como resultado, os sistemas computacionais para o apoio às atividades de CE têm migrado de sistemas individuais para sistemas integrados que passam a considerar a arquitetura geral do ambiente computacional onde estão inseridos. Como conseqüência, dois elementos são considerados fundamentais no desenvolvimento destes sistemas integrados de CE (Figura 1), a saber (MCKAY *et al.*, 1996; YOUNG *et al.*, 1998): modelos de informações e aplicações computacionais.

Os modelos de informações capturam como e onde as informações comuns a um produto são armazenadas e como as mesmas tornam-se disponíveis para as diferentes aplicações computacionais durante o ciclo de projeto e a fabricação de tal produto. Normalmente tal elemento está relacionado com a aplicação de tecnologias relacionadas com banco de dados, o que exige a criação de um modelo o mais consistente e íntegro possível que possa ser "entendido" pelas diferentes aplicações computacionais.

Por outro lado, as aplicações computacionais focalizam a captura e representação da funcionalidade específica de uma determinada atividade de projeto ou fabricação (p.ex. Projeto para a Manufatura), além de manter uma interface com o usuário final (projetista, por exemplo). As aplicações computacionais são responsáveis também pelas mudanças (recuperação ou armazenamento) dos dados nos modelos de informações e normalmente estão relacionadas com ferramentas computacionais, tais como computação gráfica, sistemas especialistas, sistemas baseados em conhecimento, etc.

Esta estrutura fornece às empresas algumas vantagens tais como, uma completa integridade nos dados, rápida flexibilidade, manutenção, independência de "vendedores" e suporte para o ciclo de vida do produto.

Contudo, ambos, os modelos de informações e as aplicações computacionais, não permanecem estáticos ao longo do tempo, estando suscetíveis a mudanças (evoluções) ou até mesmo substituições. Por esta razão, modelos conhecidos de referência devem ser adotados para a criação destes elementos visando, assim, uma maior portabilidade em tais mudanças.

Este artigo demonstra como o processo de desenvolvimento de sistemas de informações, que apóiam o projeto e manufatura integrados, pode ser modelado e conduzido através da utilização conjunta do RM-OPD (*Reference Model for Open Distributed Processing*) como modelo de referência e da UML (*Unified Modelling Language*) como notação padrão. É demonstrado como a UML pode ser utilizada como um padrão para a representação dos diferentes níveis do RM-ODP, enfocando-se nos três primeiros níveis deste modelo.

As duas próximas seções deste artigo apresentam uma breve descrição do RM-ODP e da UML. A seção 4 traz uma representação diagramática de como a UML pode ser aplicada aos três primeiros níveis de representação do RM-ODP, onde a ferramenta Rational Rose® é utilizada para o modelagem gráfica dos diagramas da UML. Para tanto um sistema de informações, chamado IMSS (*Injection Mould Support System*) é apresentado como exemplo para exposição das idéias. Por último as conclusões são apresentadas.

2. Modelo de Referência para Processamento Distribuído Aberto

O RM-ODP (ISO/IEC, 1995) foi criado para servir como modelo de referência para a descrição de sistemas distribuídos abertos e é aceito atualmente como um padrão *de facto* (BLAIR *et al.*, 1996). Cinco níveis estão definidos neste modelo e devem ser seguidos para que o desenvolvimento de um sistema de informações seja compatível com este padrão (Figura 2).

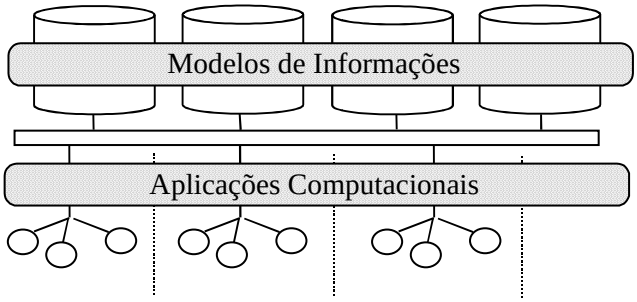


Figura 1 – Estrutura de Sistemas Integrados de CE

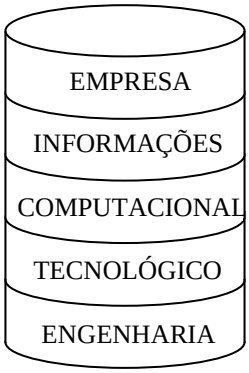


Figura 2 – Níveis de representação do RM-ODP

Estes níveis (ou vistas) podem ser encontrados em detalhes em ISO/IEC 10746-1 (ISO/IEC, 1995), contudo abaixo uma breve descrição dos mesmos é fornecida:

I. Nível de Empresa (*Enterprise Viewpoint*): descreve o sistema de informações em termos de o que o mesmo deve fazer. As necessidades e especificações administrativas e técnicas que guiam e justificam o projeto do sistema também são capturadas neste nível;
II. Nível de Informações (*Information Viewpoint*): descreve o sistema de informações em termos de estruturas de informações, fluxo de informações e restrições relacionados com a manipulação das mesmas;
III. Nível Computacional (*Computational Viewpoint*): descreve o sistema de informações em termos de operações e características computacionais do processo de mudança de informações;
IV. Nível Tecnológico (*Technological Viewpoint*): descreve o sistema de informações em termos dos componentes que o sistema é construído, e
V. Nível de Engenharia (*Engineering Viewpoint*): descreve o sistema de informações em termos de recursos de engenharia para suportar a natureza distribuída do processamento.

Apesar do RM-ODP apresentar uma descrição essencial dos conteúdos de cada nível, que deve ser considerada e assim facilitar a comparação entre sistemas alternativos, aderir ao mesmo não resolverá todas as questões práticas envolvidas no projeto e implementação de sistemas de *software*. O RM-ODP não determina como o sistema de informações deve ser projetado e implementado, bem como não define

quais as ferramentas ou técnicas que devem ser utilizadas para a representação de cada nível. Tal tarefa é deixada a critério do usuário. Por esta razão, além do RM-ODP, torna-se necessária a aplicação de métodos formais que possam guiar o projeto e implementação efetiva e consistente de sistemas de *software* através de cada um dos seus níveis.

A aplicação da tecnologia orientada a objetos tem se destacado no desenvolvimento de sistemas computacionais de informações. Os objetos podem fornecer uma representação mais realista e compatível com o mundo externo, onde cada objeto guarda sua identidade, estado e comportamento (BOOCH, 1994). Por esta razão, tal tecnologia pode fornecer uma representação potencial para os níveis do RM-ODP. Modelos para desenvolvimento de *software* orientados a objetos, como por exemplo modelo espiral (WAINWRIGHT *et al.*, 1996) e metodologias orientadas a objetos, que exploram os aspectos comportamentais dos objetos, têm sido desenvolvidas para apoiar a análise e projeto de sistemas de informações (MONARCHI & PURH, 1992; WU, 1995). Contudo um padrão comum para a representação dos elementos, e.g. objetos, que serão utilizados durante as fases de análise e implementação de um sistema de informações não tem sido claramente definido.

Métodos como diagramas de fluxo de dados e decomposição funcional (p. ex. IDEF0) têm sido usados para representar a funcionalidade de um sistema computacional, normalmente relacionadas com o primeiro nível do RM-ODP, *i.e.* Nível de Empresa (MOLINA *et al.*, 1994; MCKAY *et al.*, 1997). Contudo, tais ferramentas não são completamente orientadas a objetos, o que pode fazer suas contribuições limitadas, no que se refere a suportar o processo de migração natural das informações através das diferentes fases e perspectivas do desenvolvimento de um sistema de informações. Somado a isto, a aplicação dessas ferramentas exige uma análise detalhada das funcionalidades e necessidades do sistema já no seu início, o que não necessariamente fornece uma maior contribuição na representação das funcionalidades dos objetos. Ao contrário, fica difícil uma clara separação entre a funcionalidade necessária pelo sistema e as atividades dos usuários já existentes, o que podem, ou não, representar a forma mais efetiva do novo sistema atender as suas necessidades funcionais. Se um nível demasiado de refinamento nos detalhes de um sistema é enfocado já nos seus estágios iniciais de análise, corre-se o risco de enfocar aspectos gerais do sistema, e portanto, difícil de identificar claramente as fronteiras e características dos objetos.

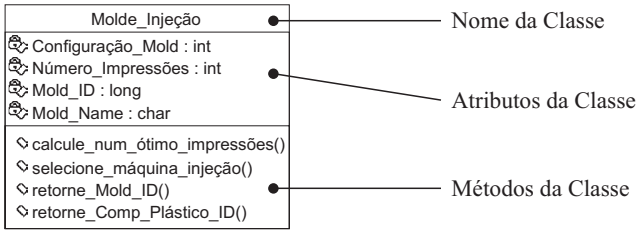
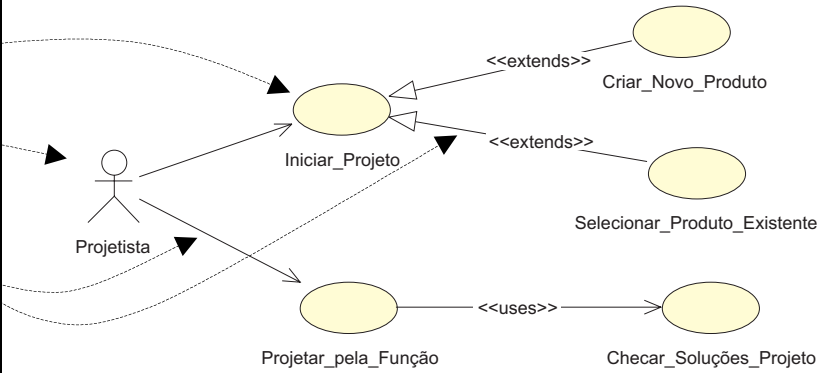
As evoluções do IDEF0, como IDEF1x, permitem uma melhor representação da funcionalidade de um sistema no Nível de Empresa (KUSIAK *et al.*, 1997). Entretanto, segundo esses autores a utilização dos objetos para a representação dos modelos de informações necessita ser ainda mais pesquisada. O IDEF4 se propõe a auxiliar a aplicação da tecnologia orientada a objetos através de um conjunto de diagramas. Contudo, este não se apresenta ainda como uma linguagem amplamente difundida no meio industrial. Somado a isto, as diferentes metodologias IDEF's ainda não permitem uma modelagem completa, integrada e que possa migrar entre as diferentes fases do processo de análise, projeto e desenvolvimento de sistemas de informações orientado a objetos.

Desta forma, uma notação o mais compatível possível para guiar as fases de análise, projeto e implementação de um sistema de informação, dentro dos diferentes níveis do RM-ODP, deveria ser adotada.

3. A Linguagem de Modelagem Unificada (UML) e *Use Cases*

A UML (*Unified Modelling Language* – Linguagem de Modelagem Unificada) surgiu, nos últimos anos, da união de métodos anteriores para análise e projeto de sistemas orientados a objetos e em 1997 passou a ser aceita e reconhecida como um padrão potencial de notação para modelagem de múltiplas perspectivas de sistemas de informações pela

Tabela 1. 1 – Descrição dos Diagramas da UML

Diagramas	***Descrição/Representação***
Representação de uma Classe ♦ *Nome* ♦ *Atributos* ♦ *Métodos*	*Representa um conjunto de objetos que compartilham os mesmos atributos, métodos, relacionamentos e semântica.* 
Use Cases ***Elementos*** ♦ *Use Cases* ♦ *Atores* ♦ *Relações: (dependência, generalização e associação)*	*Representam um alto nível de funcionalidade de um sistema ("o que o sistema deveria fazer"). Use Cases são extraídos de discussões entre usuários finais, analistas, gerentes, etc., e são complementados pelas descrições de suas ações (Cenários) e interfaces gráficas.* 

OMG ("Object Management Group") (BOOCH *et al.*, 1999). Entre os métodos que deram origem a esta linguagem de modelagem visual estão: Booch (BOOCH, 1994), OMT (*Object Modelling Technique*) e OOSE (*Object Oriented Software Engineering*). A UML define um conjunto básico de diagramas e notações que permitem representar as múltiplas perspectivas (estruturais / estáticas e comportamentais / dinâmicas) do sistema sobre análise e desenvolvimento. Dentre os diagramas podem ser citados: Diagramas de *Use Cases*, Diagramas de Classes, Diagramas de Interações (Seqüência ou Colaboração), Diagramas de Atividades e Diagramas de Estado e Transição. As Tabela 1. 1, Tabela 1. 2 e Tabela 1. 3 descrevem brevemente alguns destes diagramas. Informações complementares sobre outros tipos de representações diagramáticas da UML podem ser encontradas em (BOOCH *et al.*, 1999; JACOBSON *et al.*, 1999). O ambiente de projeto de moldes de injeção foi genericamente utilizado como exemplo para a representação de tais diagramas.

Diferente do RM-ODP, a UML oferece um suporte direto para o projeto e implementação de cada perspectiva do sistema em desenvolvimento e também uma notação para sua representação. Por esta razão, para a sua completa utilização, torna-se necessário um processo/metodologia que permita a migração e evolução das informações através das diferentes fases de representação, tais como funcionalidade, análise e projetos, implementação, etc. JACOBSON *et al.* (1999) fornecem um processo chamado Processo de Desenvolvimento de *Software* Unificado (*UML process*).

Tabela 1. 2 – Descrição dos Diagramas da UML

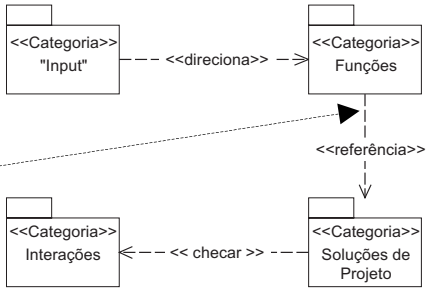
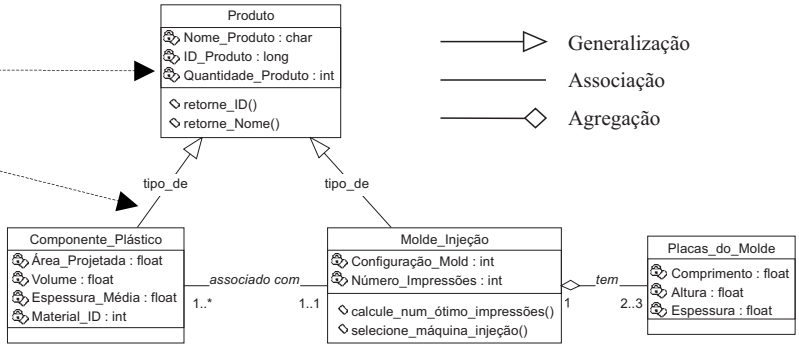
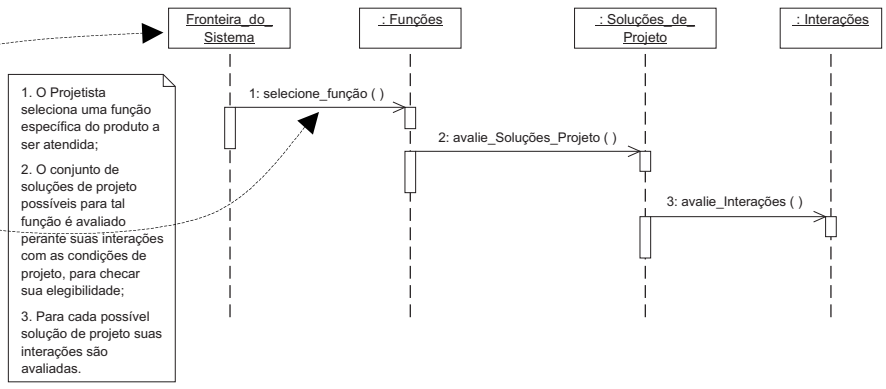
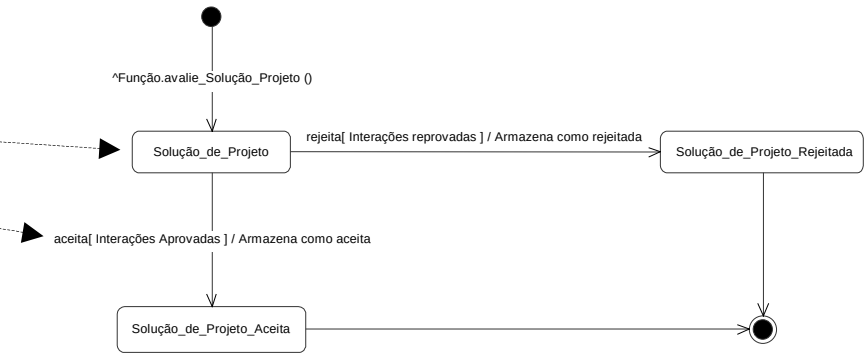
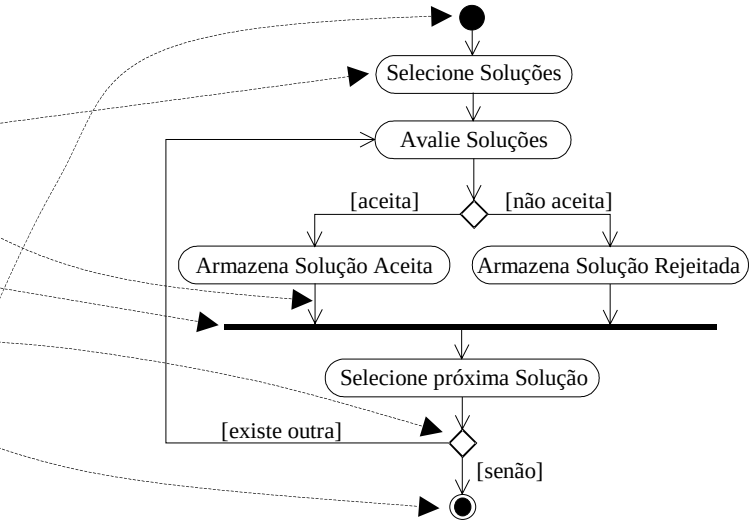
Diagramas	***Descrição/Representação***
Classes Aplicados à representação de Categorias (ou Pacotes) ***Elementos*** ♦ *Categorias/ Pacotes* ♦ *Relações: (dependência e associação)*	*Representam associações entre as principais partes do sistema (Pacotes ou Categorias). As Categorias representam um grupo de objetos com funcionalidade similar e apóiam o projeto e implementação modular do sistema em desenvolvimento. Por exemplo, Funções e Soluções de Projeto são duas categorias, sendo que a primeira é uma referência para a segunda.* 
Classes ***Elementos*** ♦ *Classes* ♦ *Relações: (generalização, associação e agregação)*	*Representam a estrutura interna (atributos e métodos) e as relações entre um conjunto de classes. Tais relações definirão principalmente a forma em que os objetos serão implementados. Por exemplo, Molde_Injeção e Componente_Plástico são tipos de Produto e assim herdam os atributos e métodos de tal classe.* 
Seqüência (ou Interação) ***Elementos*** ♦ *Classes ou Categorias* ♦ *Métodos*	*Capturam e representam a colaboração necessária entre classes, ou Categorias, através de seus métodos. Basicamente, os aspectos comportamentais dos objetos são focalizados, mostrando quais métodos são necessários para satisfazer um "Use Case" específico. Por exemplo, o "Use Case" avalie Soluções Projeto mostra como as categorias Funções e Soluções_de_Projeto colaboram para atender uma funcionalidade específica do sistema.* 

Tabela 1. 3 – Descrição dos Diagramas da UML

Diagramas	***Descrição/Representação***
Estado e Transição ***Elementos*** ♦ *Estados da classe* ♦ *Métodos*	*Representam o comportamento interno de uma classe durante sua vida, mostrando como específicos eventos (métodos) podem mudar as fases da vida de mesma. Por exemplo, após uma avaliação, uma Solução_de_Projeto poderá tornar-se Aceita ou Rejeitada.* 
Atividades ***Elementos*** ♦ *Atividade/Ação;* ♦ *Transição;* ♦ *Barra de sincronização;* ♦ *Decisão;* ♦ *Marcadores de Início e fim.*	*Representa o conjunto de passos a serem executados por um Método, mostrando como uma operação pode(ria) ser implementada.* 

TEXEL & WILLIAMS (1997) propõem um processo baseado em *Use Cases* combinado com Booch, OMT e UML, para o desenvolvimento de sistemas orientados a objetos.

Em ambos os processos, os *Use Cases* definem o primeiro nível de representação do sistema e resultam de uma fase de captura das "necessidades" a serem atendidas pelo mesmo. Os *Use Cases* representarão, num nível mais geral, as funcionalidades do sistema em desenvolvimento e guiarão todas as fases subseqüentes de análise, projeto, implementação e testes do sistema computacional.

Este artigo não tem como objetivo maior explorar o processo a ser aplicado para a modelagem das informações, visto que se trata de um outro tópico bastante abrangente. A Figura 3 mostra, de forma simplificada, como tal processo pode ser desenvolvido (TEXEL & WILLIAMS, 1997).

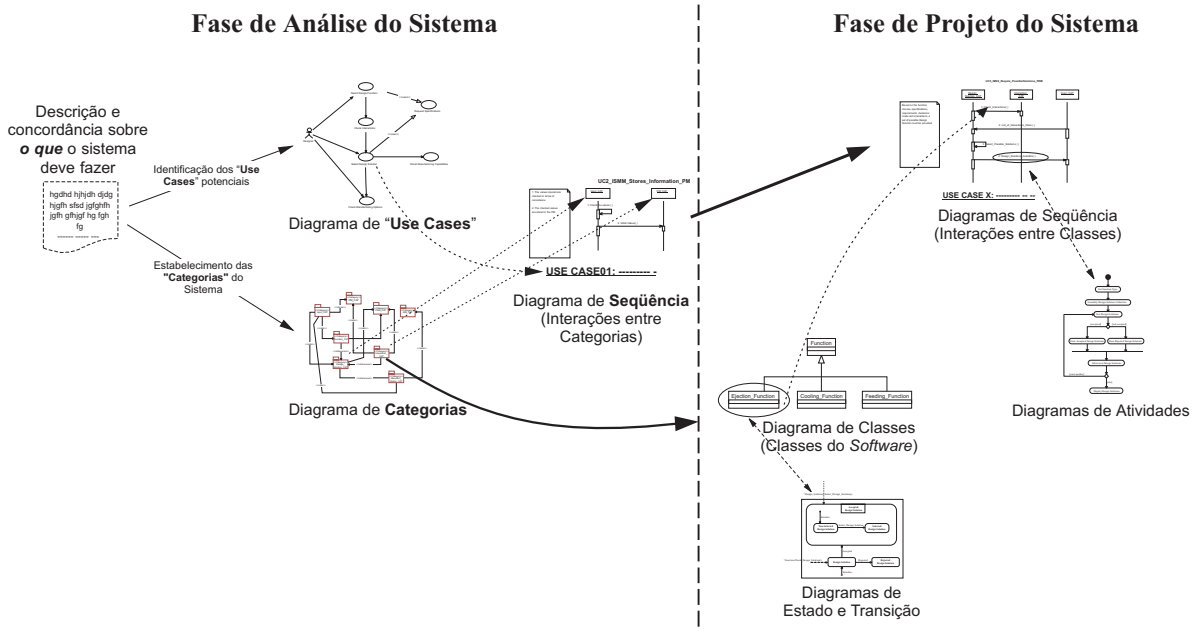


Figura 3 – Processo de Identificação dos *Use Cases*

Com base em uma descrição detalhada do sistema, principalmente enfocando as expectativas dos usuários em termos de "o que o sistema deveria fazer", *Use Cases* potenciais são extraídos, bem como as Categorias do sistema. As Categorias (ou Pacotes) são outros tipos de elementos da UML e representam os módulos principais (grupo de objetos com funcionalidade similar) do sistema em desenvolvimento. Com base nestes dois elementos, uma descrição geral de como as Categorias interagem entre si para executar cada *Use Case*, pode ser representada por diagramas de Seqüência. Esta fase é definida como análise do sistema onde tais representações podem ser utilizadas para um melhor esclarecimento e discussão com os usuários e responsáveis pela implementação do sistema. Numa fase seguinte, caracterizada com maior ênfase no projeto do sistema, busca-se um refinamento destas representações, a nível dos objetos que farão parte do sistema. Ambos os diagramas, Classes e Interações são utilizados e apoiados por representações mais detalhadas dos aspectos comportamentais dos objetos, através de diagramas de Estado e Transição e diagramas de Atividades.

4. A UML e o RM-ODP Apoiando o Desenvolvimento de Sistemas de Informações

4.1 Aplicação da UML para a Representação dos Níveis do RM-ODP

A Figura 4 mostra uma representação geral dos diagramas da UML dentro da estrutura do RM-ODP. Este artigo está concentrado apenas nos três primeiros níveis do RM-ODP uma vez que, nestes níveis, encontram-se os principais aspectos relativos à análise e projeto de um sistema de informações. *Use Cases* e Categorias aparecem como principais elementos na representação do Nível de Empresa, guiando os níveis subseqüentes através de objetos, que são os principais elementos dos níveis 2 e 3, Informação e Computacional respectivamente. No Nível de Informações são utilizados os diagramas de Classes e diagramas de Estados e Transição, enquanto que no Nível Computacional são utilizados os diagramas de Seqüência e diagramas de Atividades.

Para facilitar a apresentação das idéias propostas neste artigo, alguns exemplos de diagramas UML foram desenvolvidos. Estes exemplos

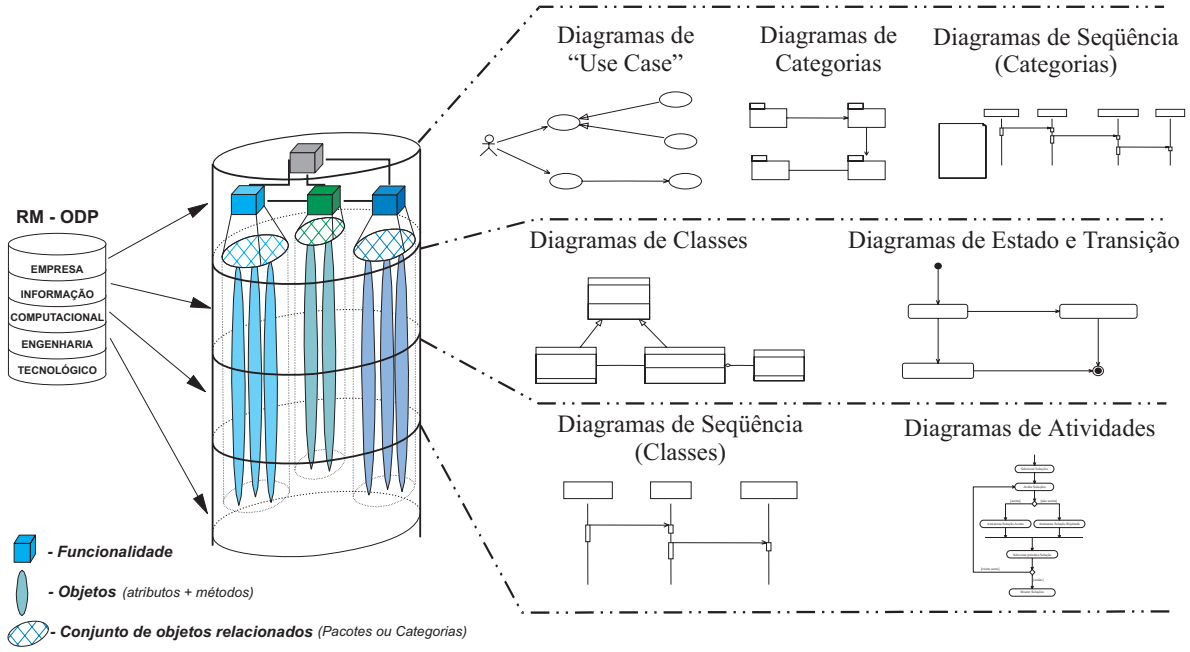


Figura 4 – Notação UML aplicada aos 3 primeiros níveis do RM-ODP

são baseados em um projeto de pesquisa na área de reutilização de informações no projeto de moldes de injeção, onde um modelo de informações chamado "Product Range Model" (*Modelo Variável do Produto*) foi definido (COSTA & YOUNG, 1998). Tal projeto não será descrito em detalhes neste artigo, sendo utilizado somente como um sistema exemplo para a demonstração do uso combinado do RM-ODP e da UML.

A pesquisa relacionada com o *Modelo Variável do Produto* segue a mesma abordagem do projeto de pesquisa MOSES (ELLIS *et al.*, 1995), desenvolvido na Universidade de Loughborough (UK), onde modelos de informações implementados em banco de dados fornecem todas as informações solicitadas pelas várias aplicações computacionais utilizadas durante o processo de projeto e manufatura do produto.

Um sistema chamado IMSS (*Injection Mould Support System*) é usado aqui como um exemplo de sistema de informações. Este sistema inclui dois modelos de informações, a saber Modelo do Produto e o *Modelo Variável do Produto*, e aplicações de *software* que usam as informações capturadas por aqueles modelos, para apoiar o projeto funcional de moldes de injeção (Figura 5). O *Modelo Variável do Produto* armazena as relações entre as funções do molde e suas soluções potenciais de projeto, e as relações entre estas soluções e suas interações de projeto, chamadas simplesmente de interações. Tais interações relacionam-se com as informações sobre o produto molde de injeção, que estão armazenadas no Modelo do Produto.

Para a demonstração neste artigo, uma ênfase maior foi dada aos aspectos relacionados com definição da estrutura de informações para o *Modelo Variável do Produto*, bem como para as relações entre os seus elementos.

4.2 *Use Cases* e Categorias – Nível de Empresa do RM-ODP

O Nível de Empresa do RM-ODP captura os principais objetivos e restrições do sistema em desenvolvimento. Desta forma, os *Use Cases*, que representam a funcionalidade do sistema ("*o que o sistema deveria fazer*"), enquadram-se como um importante elemento para representação

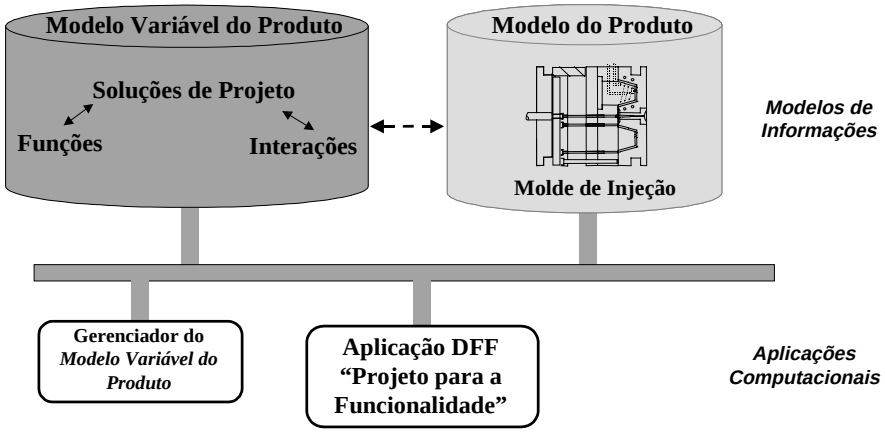


Figura 5 – Ambiente do IMSS

deste nível. Embora os *Use Cases* tentem capturar o que os usuários querem do sistema, eles não especificam como o sistema deveria ser. Isso é definido nas fases seguintes de projeto do sistema. Os *Use Cases* por si só não apresentam muitas informações, e assim os *Cenários dos Use Cases* complementam suas descrições. Estes *Cenários* são compostos por fluxos de eventos, as ações e reações do *software*, as restrições, necessidades de interfaces gráficas, etc. Esta descrição dos *Use Cases*, através de seus *Cenários*, fornecem informações úteis também para a especificação das propriedades (atributos e métodos) das classes necessárias para executar os *Use Cases*.

Apesar de permitir uma comum representação da funcionalidade do sistema computacional em questão, somente com a identificação inicial mais clara da estrutura geral do sistema poder-se-á alcançar um melhor entendimento e estabelecimento das responsabilidades das equipes envolvidas no seu desenvolvimento.

Uma das principais transições na utilização da notação UML está em através dos *Use Cases* representar o sistema em termos de objetos. TEXEL & WILLIAMS (1997) mostram este processo por intermédio de uma metodologia em que uma lista das Categorias potenciais são extraídas.

A identificação das Categorias do sistema é recomendada para uma definição clara dos módulos que farão parte do mesmo. Assim, a adoção das representações diagramáticas de *Use Cases* e Categorias do sistema fornecerão uma documentação compreensiva que servirá como base para discussões entre gerentes, usuários finais, programadores, etc.

A Figura 6 mostra uma representação simplificada de alguns dos *Use Cases* e Categorias identificados para o caso em aplicação, i.e. ISMM. No exemplo citado, um dos *Use Cases* é "Avaliar_Soluções_Projeto", que está relacionado com duas categorias principais, *Funções e Soluções de Projeto*, sendo que a primeira Categoria é utilizada como elemento de referência para a segunda. Por exemplo, para a função do molde "extrair componente plástico", estará associado um possível conjunto de soluções, tais como pinos de extração, placas de extração, etc.

A representação do relacionamento destas Categorias com os *Use Cases* que farão uso das mesmas é feita através dos diagramas de Seqüência. Enquanto os Diagramas de Categorias representam as associações entre as Categorias fornecendo uma visão geral das relações entre as principais estruturas do sistema, os Diagramas de Seqüência representam as relações particulares entre as Categorias, para cada *Use Case*. Tais representações fornecem para a equipe de desenvolvimento, um "feedback" em termos do projeto inicial do sistema. Estas representações guiarão os futuros estágios de análise e projetos do sistema.

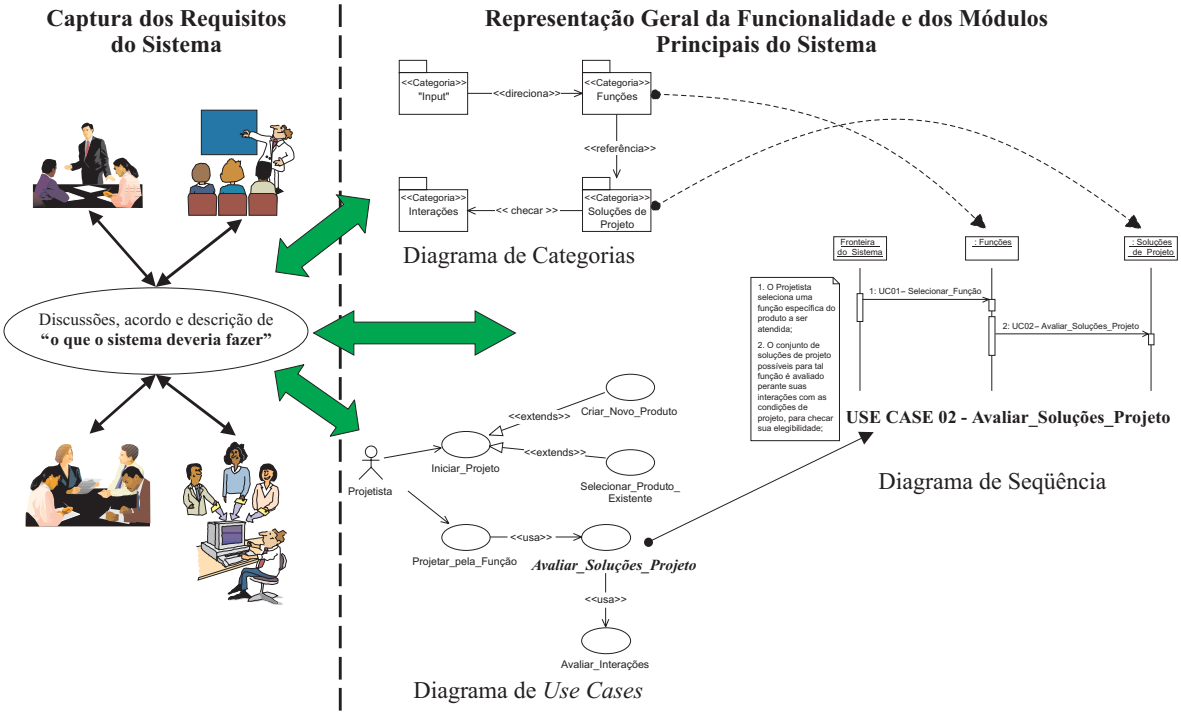


Figura 6 – Identificação dos *Use Cases* e Categorias – 1º Nível do RM-ODP

4.3 UML e a Representação dos Objetos – Níveis de Informação e Computacional do RM-ODP

Uma vez que o Nível de Empresa tenha concordância suficiente sobre a funcionalidade e estrutura geral do sistema, o próximo passo é a identificação dos objetos que farão parte deste sistema. Assim, as propriedades e comportamentos dos objetos são analisados e desenvolvidos, bem como as associações entre este objetos, através de suas agregações e hierarquias. Para apoiar o processo de definição dos objetos, os *Cenários* de cada *Use Case* devem ser melhor explorados e esclarecidos.

Nesta fase, onde uma estrutura mais detalhada do sistema é representada, os aspectos relativos a plataforma utilizada para implementação passam a ser considerados, permitindo assim um maior refinamento na definição dos atributos e métodos dos objetos. Por exemplo, no caso de modelagem das informações sobre produto, o uso de um banco de dados orientado a objetos puro ou o uso de um banco de dados relacional pode resultar em diferentes considerações sobre as formas de representação e relacionamentos dos objetos. Contudo, indiferente da plataforma escolhida, o resultado final desta fase deverá ser uma clara representação de cada objeto, fornecendo assim, para a equipe envolvida na programação do sistema, uma clara visão de todos os detalhes de tal objeto.

Como apontado na seção 1 deste artigo, dois elementos principais estão presentes no desenvolvimento de sistemas integrados para o apoio às atividades de projeto e fabricação de peças: os modelos de informações e as aplicações computacionais. Tais elementos estão relacionados com diferentes aspectos dos objetos. Enquanto as estruturas definidas para a representação dos modelos de informações estarão primariamente relacionadas com as informações a serem armazenadas na base comum de dados, por exemplo os valores dos atributos de cada objeto ou suas relações com outros objetos, as aplicações computacionais,

estarão mais relacionadas com o aspecto comportamental dos objetos, ou seja, seus métodos. Tais características determinam a ênfase na representação de um objeto no segundo ou terceiro níveis do RM-ODP.

4.3.1 Objetos e o Nível de Informações do RM-ODP

O Nível de Informações do RM-ODP está relacionado principalmente com o modelo de informações a ser utilizado pelo sistema computacional. Tal nível deve capturar uma representação o mais consistente possível da estrutura de informações, que representará este modelo, e que deverá ser "entendida" por outras aplicações computacionais possíveis.

O EXPRESS/STEP tem sido reconhecido como uma linguagem padrão para a representação do Nível de Informações do RM-ODP (MOLINA *et al.*, 1994; MCKAY *et al.*, 1997). Contudo, com o crescente uso das técnicas orientadas a objetos, e mais recentemente a UML, conversões entre ambas as linguagens tem se tornado mais comuns (GHODOUS & VEORPE, 1998). Somado a isto, considerando que o sistema final a ser desenvolvido usará métodos orientados a objetos, o uso de notações padrões nesta área tem se tornado significativamente vantajosos. Desta forma, os diagramas de Classes da UML podem representar de forma adequada este nível do RM-ODP.

Os Diagramas de Classes permitem uma representação da estrutura interna de cada objeto, bem como os relacionamentos entre os objetos, fornecendo uma consistente representação da estrutura (ou esquema) do banco de dados que conterá tais informações. Embora tal diagrama mostre também os métodos associados a cada objeto, os mesmos não possuem significante importância para este nível de representação.

A Figura 7 mostra o segundo nível do RM-ODP, i.e. Nível de Informações, dentro do exemplo do IMSS, onde uma Categoria é representada em termos de Classes, com seus atributos e relações. São mostrados diferentes tipos de soluções de projetos (p.ex. Resfriamento, Extração, etc.) que podem fazer parte de um molde de injeção. Apesar de todas estas soluções serem tipos de *Soluções de Projeto* cada uma possuirá atributos particulares, que justificam a definição de diferentes tipos classes. Por exemplo, enquanto para uma solução de resfriamento o diâmetro e posicionamento dos canais de refrigeração bem como a temperatura do líquido refrigerante são aspectos importantes, para uma solução de extração, além do posicionamento e dimensões, o curso de extração passa também a ser um importante atributo.

Os Diagramas de Estado e Transição complementam a representação interna do comportamento dos objetos, mostrando as diferentes fases de seu ciclo de vida e os eventos específicos (métodos) que podem causar a transição de um estado para outro. Os estados são definidos como situações, durante a vida de um objeto, em que ele satisfaz alguma condição, executa alguma atividade ou aguarda algum evento. A transição é a relação entre dois estados de um objeto indicando que, baseado em certas ações e satisfação de uma condição específica, o estado de tal objeto pode mudar para um segundo estado. Por exemplo, na Figura 7, dependendo das condições/especificações durante o projeto de um molde, uma solução de projeto pode ser considerada aceita ou rejeitada.

4.3.2 Objetos e o Nível Computacional do RM-ODP

O Nível Computacional do RM-ODP está relacionado com os processos de mudança dos dados nos modelos de informações, e captura os aspectos dinâmicos do sistema em desenvolvimento, ou seja, a representação das operações responsáveis por tais mudanças.

Os Diagramas de Seqüência fornecem uma representação de como os objetos podem interagir, por meio de seus métodos, para realizar cada uma das funcionalidades específicas do sistema, i.e., *Use Cases*. Tais diagramas representam, também, as seqüências de interação

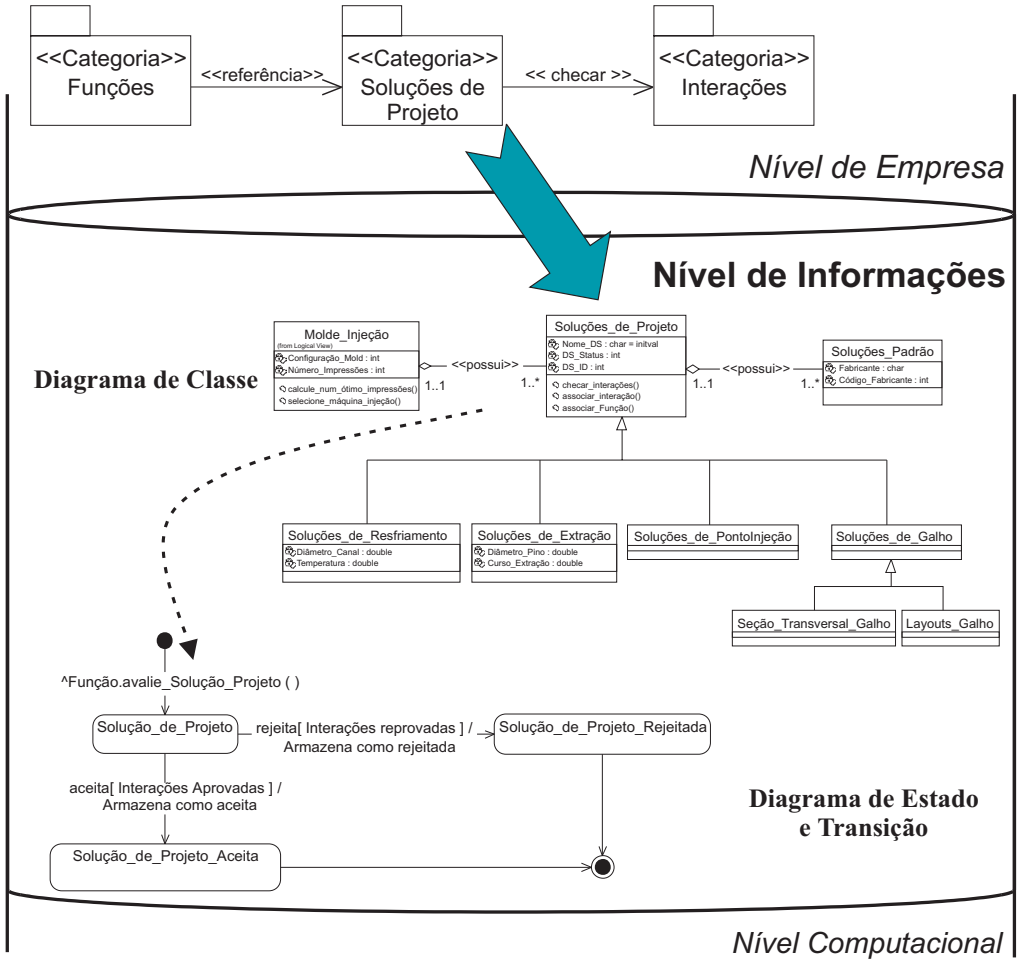


Figura 7 – Diagramas da UML no 2º Nível do RM-ODP

de cada objeto, capturando os aspectos relacionados com o comportamento de cada objeto.

Cada um dos Diagramas de Seqüência, que inicialmente representavam as interações gerais entre as Categorias do sistema, são refinados a um nível de representação dos métodos específicos de cada objeto envolvido para implementação de cada *Use Case*.

Os Diagramas de Atividades são utilizados neste nível de representação para detalhar os passos (ou atividades), dentro de um processo computacional, a serem executados para a realização de cada método, fornecendo uma representação de o que deve ser capturado pela implementação de tal método. Desta forma tais diagramas estão relacionados com a representação dos aspectos dinâmicos do sistema. Cada atividade resulta em alguma ação que, por sua vez, resulta em trocas no estado do sistema ou retorno de algum valor. Ações podem incluir a chamada de outras operações, envio de sinais, criação de um objeto ou ainda alguma expressão puramente computacional.

A Figura 8 mostra o Nível Computacional do RM-ODP, para o exemplo do IMSS, onde cada função principal do sistema, i.e. *Use Case*, é representado em termos de métodos e operações específicas, necessárias entre as classes, para sua implementação. Assim, enquanto a representação geral do relacionamento entre as Categorias *Funções* e *Soluções de Projeto*, no Nível de Empresa do RM-ODP, era através de um método "*avaliar_soluções_projeto*", tal representação em termos de classes, passa a ser mais detalhada,

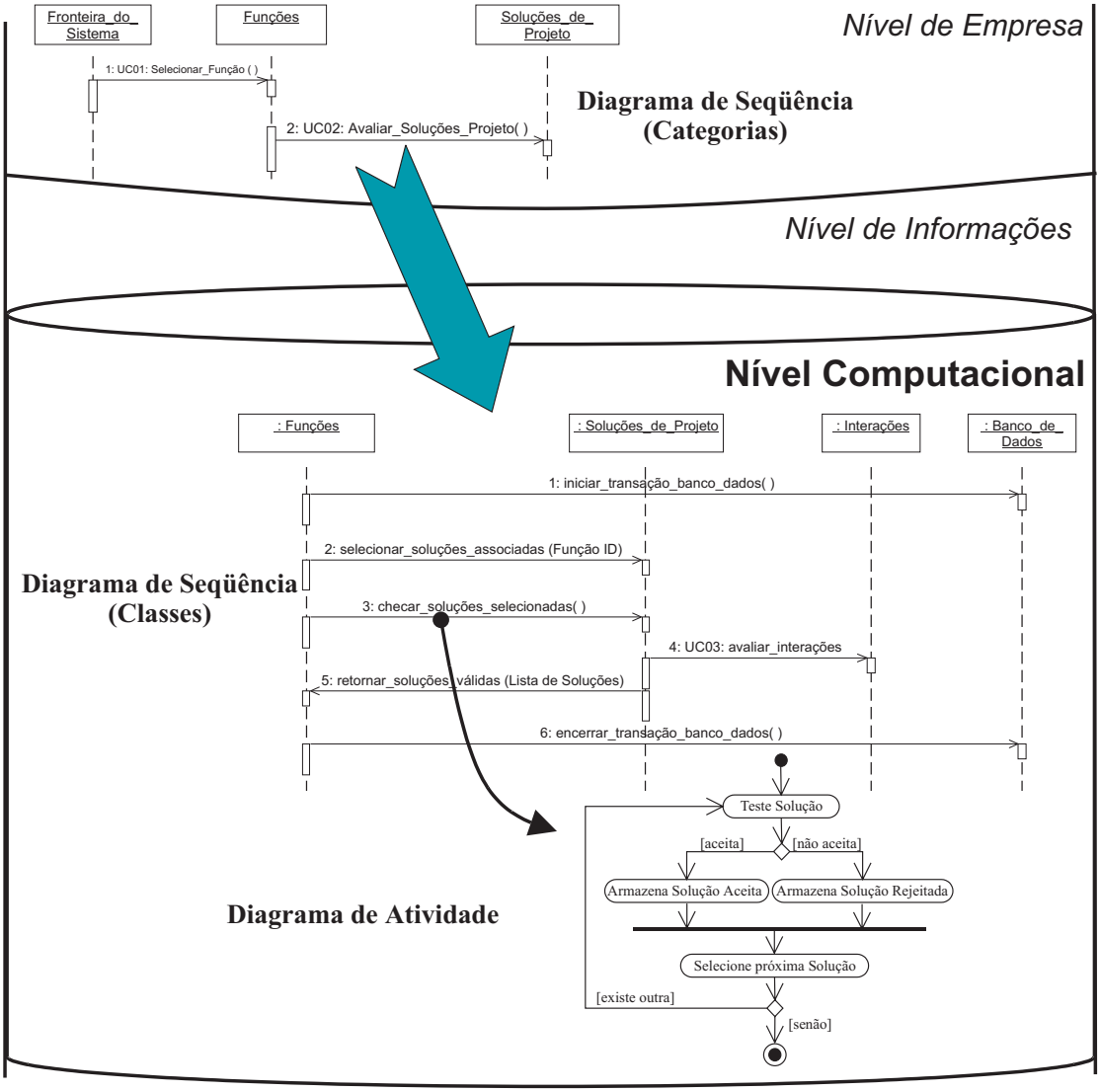


Figura 8 – Diagramas da UML no 3º Nível do RM-ODP

considerando aspectos como abrindo e fechando transações com o banco de dados, decidir quando e como uma solução de projeto é aprovada ou não, etc.

No exemplo do IMSS, para checar quais soluções de projeto são aceitas, ou não, para uma condição de projeto específica, um conjunto de soluções de projeto associadas com uma determinada função do molde devem ser selecionadas e para cada uma suas interações devem ser verificadas. O resultado deste processo de verificação para cada solução de projeto é retornado para a classe *Funções*. Adicionalmente, o método número 3, i.e. *checar_soluções_selecionadas()*, é representado, através de diagramas de Atividades, onde os critérios para aceitar ou rejeitar uma solução de projeto são definidos.

5. Implementação do IMSS

As representações fornecidas nos Níveis de Informações e Computacional do RM-ODP deveriam ser suficientemente detalhadas para apoiar as fases seguintes de desenvolvimento do sistema de informações, i.e. programação do sistema.

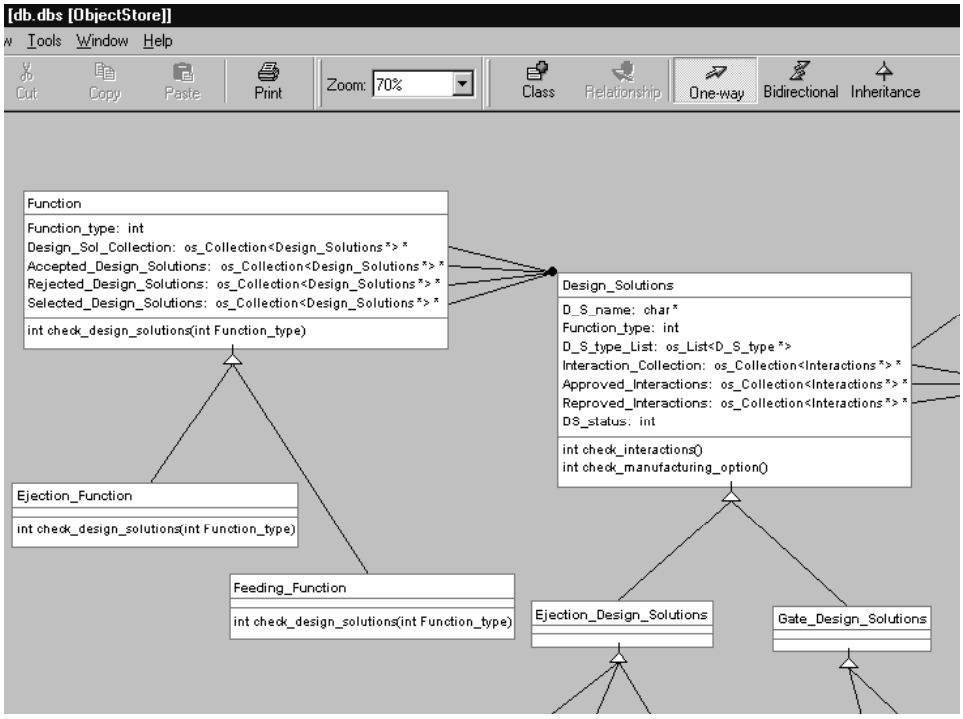


Figura 9 – Exemplo do esquema do banco de dados para o ObjectStore®

No caso do IMSS, uma implementação de tal sistema foi realizada baseado nas representações fornecidas pelos três primeiros níveis do RM-ODP. Embora o Nível de Tecnologia do RM-ODP não tenha sido definido dentro do escopo deste artigo, para efeitos de demonstração um banco de dados orientado a objetos (ObjectStore®) e um ambiente de programação (Visual C++®) foram utilizados. Enquanto o ObjectStore® propiciou a captura da representação do modelo de informações persistente, o Visual C++® propiciou um bom suporte em termos de visualização dos resultados. Particularmente para o caso do ObjectStore®, foi possível a utilização dos diagramas de classes, na notação UML, para a geração da estrutura de informações persistentes do sistema, i.e. esquema do banco de dados.

A Figura 9 mostra um exemplo de representação de parte do esquema do banco de dados criado para o IMSS, dentro do ambiente ObjectStore®. Neste caso são enfocadas as classes Funções (*Functions*) e Solução de Projetos (*Design Solutions*), bem como suas relações.

A Figura 10 mostra alguns exemplos destas duas classes citadas acima, onde as funções específicas de um molde de injeção, i.e. *Feed Impression* (alimentar cavidade), são associadas com soluções de projetos particulares. Ambas as figuras, 9 e 10, mostram o resultado da representação do Nível de Informações para o exemplo do IMSS, que neste caso, foi implementado através da tecnologia de banco de dados orientado a objetos.

A Figura 11 mostra uma das telas do sistema IMSS, onde é enfocado o resultado de um processo de pesquisa por soluções de projeto que atendam a função "Distribuir os Canais de Alimentação" (*Distribute Runner Layout*). Neste caso é mostrado os conjuntos de soluções de projetos "Aceitas" (*Accepted*) e Rejeitadas (*Rejected*), conforme as especificações anteriores de projeto, que são mostradas na janela de diálogo a direita. Neste caso são demonstrados alguns dos resultados da representação do Nível Computacional do RM-ODP para o IMSS, através da implementação dos métodos dos objetos. Tais métodos permitem definir quais

	Function_Name	DS_Name
1	Feed Impression	Sprue Gate
2		Rectangular Gate
3		Diaphragm Gate
4		Tunnel/Submarine Gate
5		Pin Point Gate
6		Hot Point Gate
7	Eject Rib	Normal Ejector Pin
8	Distribute Impressions	One Impression Layout
9		Circular Impression Distribution
10		In Line Impression Distribution
11		Matrix Impression Distribution
12	Eject Impression - by Wall	Normal Ejector Pin
13		D-Shaped Ejector Pin
14		Sleeve Ejector Pin
15		Stripper Ring
16		Stripper Plate
...		
27	Distribute Runner Layout	Circular Spoke Layout
28		Circular Y-Layout
29		Rectangular S-Layout
30		InLine Unbalanced - T-Layout
31		Rectangular H-Layout
32		Rectangular X-Layout
33		Rectangular Y-Layout
34		Rectangular Unbalanced - H-Layout
35		Hot Runner Manifold - HR2
36		Hot Runner Manifold - HRST4
37		Hot Runner Manifold - HRSQ4
...		

Figura 10 – Exemplos dos objetos Funções e Soluções de Projetos

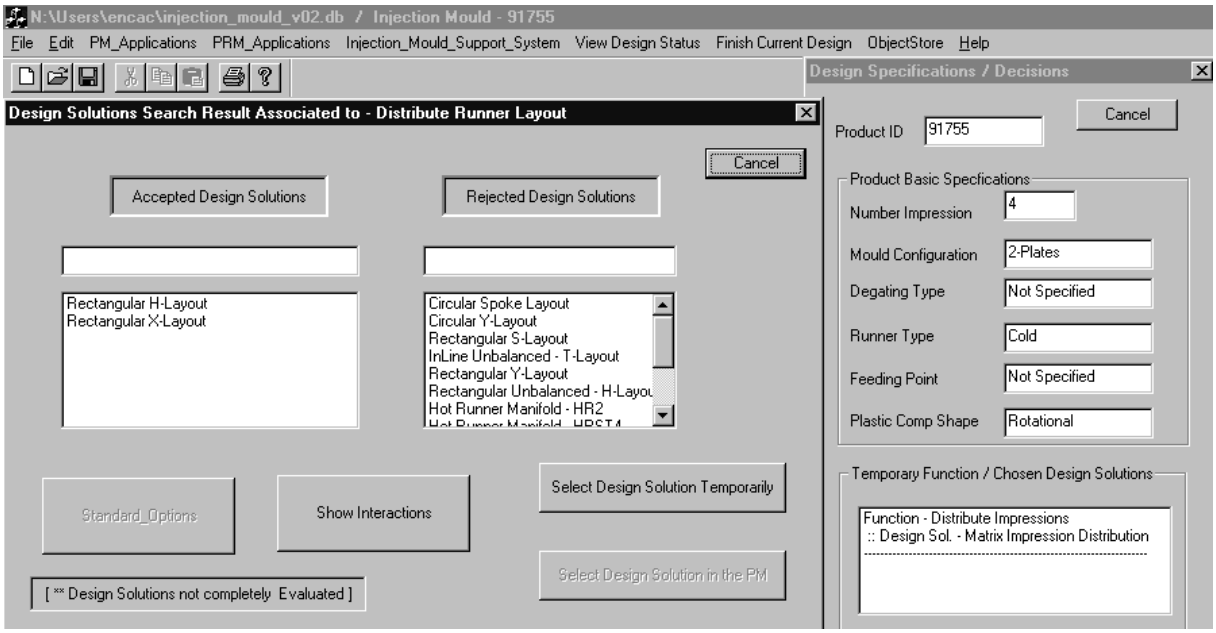


Figura 11 – Tela do IMSS: resultado da pesquisa por função de um molde de injeção

conjuntos de soluções de projeto válidas, ou não, podem ser oferecidas para o usuário. Nesta figura é mostrado também um resultado do Nível de Informações do RM-ODP, para o IMSS, através dos diferentes estados das soluções de projetos, i.e. aceitas ou rejeitadas.

6. Conclusão

A aplicação de um modelo de referência, como o RM-ODP, é considerada como um requisito mínimo para as empresas que pretendem desenvolver e trabalhar com ambientes computacionais integrados. Tal modelo permite uma maior portabilidade e integração dos sistemas que farão parte deste ambiente compartilhado de informações.

Este artigo demonstrou como este modelo de referência pode ser utilizado em harmonia com a Linguagem de Modelagem Unificada (UML) para o suporte ao desenvolvimento de sistemas de informações. Enquanto o RM-ODP oferece uma estrutura com cinco níveis para descrever um sistema de informações, a notação da UML oferece uma linguagem consistente para apoiar a representação destes diferentes níveis.

Foi mostrado também como a notação UML fornecer uma vantagem no processo de migração entre os três primeiros níveis do RM-ODP, através do uso de elementos comuns, i.e. *Use Cases*, Categorias e Classes.

Além das vantagens propiciadas no desenvolvimento de sistemas de informações, tal combinação pode servir como uma forma de comunicação entre usuários e desenvolvedores de *software*, permitindo um entendimento maior para ambos. Apesar de não ter sido explorado neste artigo, a aplicação de tal notação também permite um melhor reutilização de Classes e Categorias que já tenham sido definidas para outras aplicações computacionais.

Apesar das vantagens oriundas de tal combinação, algumas limitações podem ser identificadas em termos de fornecer uma clara interpretação do sistema para os usuários finais. Esta interpretação pode ser melhor fornecidas com o apoio de metodologias como IDEF0, que continuam tendo uma importância fundamental na representação da funcionalidade e atividades de um sistema. Somado a isto, pesquisas adicionais são necessárias para verificar como tal metodologia, i.e. IDEF0, pode ser utilizada em conjunto com a abordagem apresentada para apoiar a descrição dos *Cenários* de *Use Cases*.

Agradecimentos

Ao Governo Brasileiro (CNPq – Conselho Nacional de Desenvolvimento Científico e Tecnológico) e a Universidade de Caxias do Sul que suportaram financeiramente as atividades de doutorado do Prof. Carlos Alberto Costa na Universidade de Loughborough (Inglaterra). Aos revisores da Revista G&P pelas suas sugestões para melhoria do artigo.

Referências Bibliográficas

BLAIR, G.; COULSON, G. & DAVIES, N.: "Standards e platforms for open distributed processing". *Electronics & Communication Engineering Journal*, (June): 123-133, 1996.

BOOCH, G.: *Object-oriented analysis e design with applications.* California, The Benjamin/Cummings Publishing Company, Inc., 1994.

BOOCH, G.; RUMBAUGH, J. & JACOBSON, I.: *The Unified Modelling Language User Guide*, Addison Wesley Longman, Inc., 1999.

COSTA, C.A. & YOUNG, R.I.M.: "Product Range Models: linking functional design to the reuse of manufacturing information". *Engineering Design Conference'98 – Design Reuse*, Brunel, Professional Engineering Publishing Ltd, 1998.

ELLIS, T.I.A.; MOLINA, A.; YOUNG, R.I.M. & BELL, R.: "An information sharing platform for Concurrent Engineering". *Integrated Manufacturing Systems Engineering*, Chapman & Hall, pp. 262-275, 1995.

GHOUDOUS, P. & VEORPE, D.: "A systematic approach for product e process data modeling based on the Step standard". *Computer-Aided Civil e Infrastructure Engineering*, 13: 189-205, 1998.

ISO/IEC: "Information technology-basic reference model of open distributed processing". *British Standard Implementation of ISO/IEC 10746-1*, 1995.

JACOBSON, I.; BOOCH, G. & RUMBAUGH, J.: *The Unified Software Development Process*, Addison Wesley Longman, Inc., 1999.

KUSIAK, A.; LETSCHE, T. & ZAKARIAN, A.: "Data modelling with IDEF1x." *International Journal of Computer Integrated Manufacturing*, **10**(6): 470-486, 1997.

MCKAY, A.; BLOOR, M.S. & DE PENNINGTON, A.: "A Framework for product data." *IEEE Transactions on Knowledge e Data Engineering*, **8**(5): 825-837, 1996.

MCKAY, A.; BLOOR, M. S. & DE PENNINGTON, A.: "Realising the Potential of Product Data Engineering". *5th International Conference on FACTORY 2000*, Cambridge, UK, IEE, 1997.

MOLINA, A.; ELLIS, T.I.A.; YOUNG, R.I.M. & BELL, R.: "Methods e Tools for Modelling Manufacturing Information to Support Simultaneous Engineering". *Intelligent Manufacturing Systems Workshop*, Viena, 1994.

MONARCHI, D.E. & PURH, G.I.: "A Research Typology for Object-Oriented Analysis e Design." *Communications of the ACM*, **35**(9): 35_47, 1992.

TEXEL, P. & WILLIAMS, C.: *Use Cases Combined with BOOCH/OMT/UML: process e products*, Prentice Hall, Inc., 1997.

WAINWRIGHT, C.E.R.; LEUNG, A.C.K. & LEONARD, R.: "Objetc-oriented software development: a case study". *Computer Integrated Manufacturing System*, **9**(4):245-255, 1996.

WU, B.: "Object-oriented systems analysis and definition of manufacturing operations". *International Journal of Production Research*, **33**(4): 955-974, 1995.

YOUNG, R.I.M.; CANCIGLIERI-JNR, O. & COSTA, C.A.: "Information Interactions in Data Model Driven Design for Manufacture". *Globalization of Manufacturing in the Digital Communications Era of the 21st Century: Innovation, Agility, and the Virtual Enterprise.* Jacucci, G.; Olling, G.J.; Preiss, K.; Wozny, M., Trento, Kluwer Academic Publishers. 313-324, 1998.

THE APPLICATION OF UML TO SUPPORT COMPUTATIONAL SYSTEMS DESIGN WITHIN A REFERENCE MODEL FRAMEWORK

Abstract

The development of information systems to support design and manufacturing activities should follow a reference model in order to be compatible with major systems architectures. RM-ODP (Reference Model Open Distributed Processing) provides five level viewpoints against which information systems development can be compared and classified. The RM-ODP does not dictate how the information system should be designed and implemented. Rather it highlights the content of the essential views of the system, which must be considered and hence facilitates comparison of alternatives systems. In contrast, computational methodologies provide ways to design and build information systems but usually do not take reference models into consideration. This paper shows how reference models and computational methodologies can be used in harmony, and demonstrates this through the application of a Use Case and UML combined methodology across the RM-ODP viewpoints. Injection mould design is used as an example to the UML representation diagrams.

Key words: information modelling, RM-ODP, UML.