Comparando Modelos Arquiteturais de Sistemas Legados para Apoiar a Criação de Arquiteturas de Referência de Domínio

Aline P. V. de Vasconcelos[1,2], Guilherme Z. Kümmel[1], Cláudia M. L. Werner[1]

[1]COPPE/UFRJ – Programa de Engenharia de Sistemas e Computação
Caixa Postal: 68511 – CEP: 21941-972 – Rio de Janeiro – RJ – Brasil

[2]CEFET Campos (Centro Federal de Educação Tecnológica de Campos)
Rua Dr. Siqueira, 273 – Pq. Dom Bosco – CEP: 28030-130 - Campos dos Goytacazes-RJ - Brasil

{aline,kummel,werner}@cos.ufrj.br

Abstract. *Large organizations usually have legacy systems that represent effort and resources invested, besides encompassing rich business knowledge. They frequently develop systems of the same domain, what has been motivating the migration to reuse approaches such as Domain Engineering (DE) and Product Line (PL). Within a certain domain, a reference architecture (DSSA) represents the basis for application instantiation. In this context, legacy systems are essential information sources for domain specification. Therefore, this paper presents an approach for legacy system architectures comparison to support the creation of DSSAs, which can be used in the context of DE and LP.*

Resumo. *Grandes empresas costumam possuir sistemas legados que representam esforço e recursos investidos e embutem rico conhecimento sobre o negócio. Em geral, elas desenvolvem sistemas no mesmo domínio, motivando a migração para abordagens de reutilização como Engenharia de Domínio (ED) e Linha de Produtos (LP). Num dado domínio, a arquitetura de referência (DSSA) representa a base para a instanciação de aplicações. Nesse contexto, os sistemas legados são fontes de informação essenciais para a especificação do domínio. Assim, este artigo apresenta uma abordagem de comparação de arquiteturas de sistemas legados para apoiar a criação de DSSAs, as quais podem ser utilizadas no contexto da ED e LP.*

1. Introdução

A maioria das organizações que desenvolve software costuma construir sistemas de software em um domínio de aplicação particular, repetidamente entregando variantes de produtos pela adição de novas características (SUGUMARAN *et al.*, 2006). Dessa forma, é comum que essas organizações estejam migrando para abordagens de Engenharia de Domínio (ED) (PRIETO-DIAZ & ARANGO, 1991) e Linha de Produtos (LP) (LEE *et al.*, 2002). Uma vez que as arquiteturas de referência de domínio representam o elemento-chave para uma abordagem de ED ou LP de sucesso, a construção desses artefatos merece atenção nesse processo de migração.

As arquiteturas de referência de domínio representam a base para a instanciação de aplicações, reutilização e adaptação dos artefatos de domínio produzidos. Elas são o elemento central das DSSAs (*Domain Specific Software Architectures*), que, segundo XAVIER (2001), são mais do que uma arquitetura para um determinado domínio de aplicações, mas sim uma coleção de elementos arquiteturais especializados para um determinado tipo de tarefa (domínio) e generalizados para que seja possível seu uso efetivo através do domínio. Vale ressaltar, entretanto, que neste trabalho os termos arquitetura de referência de domínio e DSSA são utilizados como sinônimos. Tanto na ED quanto na LP, as arquiteturas de referência de domínio devem representar os elementos arquiteturais do domínio, seus relacionamentos, semelhanças e diferenças.

A fim de apoiar a criação dessas arquiteturas de referência de domínio, sistemas legados disponíveis para o domínio podem ser analisados. Grandes empresas costumam possuir uma base de sistemas legados que representam esforço e recursos investidos no passado, além de embutirem conhecimento sobre o negócio que muitas vezes não pode ser obtido de nenhuma outra fonte de informação, o que motiva a sua evolução contínua e reutilização em novos esforços de desenvolvimento. Entretanto, as abordagens de ED e LP existentes não costumam prover processos sistemáticos com técnicas de apoio à análise de um conjunto de sistemas legados no domínio, a fim de detectar as suas semelhanças e diferenças (ex: FORM (KANG *et al.*, 2002), CBD-Arch-DE (BLOIS, 2006), KobrA (ATKINSON *et al.*, 2002) e PLUS (GOMAA, 2004)). Algumas abordagens, como a de (GOMAA, 2004), comentam a importância de se extrair e realizar a consistência de modelos de aplicações do domínio, mas não indicam como fazê-lo.

Nesse contexto, este artigo apresenta uma abordagem de comparação de modelos arquiteturais de sistemas legados para apoiar a criação de arquiteturas de referência de domínio, que podem ser utilizadas no contexto da ED e da LP, denominada ArchToDSSA (KÜMMEL, 2007). ArchToDSSA está inserida em um contexto de trabalho mais amplo, sendo parte da abordagem LegaToDSSA (VASCONCELOS, 2007), a qual envolve uma fase de engenharia reversa para a recuperação das arquiteturas dos sistemas legados, i.e. ArchMine, e uma fase de comparação das arquiteturas recuperadas para a detecção das suas semelhanças e diferenças, e criação da DSSA, i.e. ArchToDSSA. A fase de engenharia reversa é importante porque, em geral, a documentação dos sistemas legados não se encontra atualizada em relação ao código.

ArchToDSSA também está inserida no contexto do projeto Odyssey (ODYSSEY, 2007), que visa o desenvolvimento de um ambiente de reutilização baseado em modelos de domínio de mesmo nome, i.e. o ambiente Odyssey. Dessa forma, ela adota a notação Odyssey-FEX (OLIVEIRA, 2006) para a representação das semelhanças e diferenças do domínio. Segundo OLIVEIRA (2006), as semelhanças e diferenças do domínio podem ser expressas através das suas variabilidades e opcionalidades. As variabilidades do domínio envolvem elementos arquiteturais invariantes, i.e. que não podem ser modificados na instanciação de aplicações, e elementos arquiteturais que representam pontos de variação (VPs), que podem ser configurados através da seleção de uma ou mais variantes. As opcionalidades do domínio são expressas através de elementos arquiteturais opcionais, i.e. que podem ou não ser selecionados na instanciação de aplicações, e de elementos arquiteturais mandatórios, que devem necessariamente ser selecionados na instanciação de aplicações

do domínio. As propriedades de variabilidade e opcionalidade são ortogonais na Odyssey-FEX, o que significa que um elemento que representa um VP, invariante ou variante pode ser ao mesmo tempo mandatório ou opcional.

Em relação à comparação dos modelos para a detecção das suas semelhanças e diferenças, elementos equivalentes entre modelos de diferentes sistemas podem estar descritos de forma diferente, possuindo, por exemplo, nomes diferentes, embora sendo semanticamente equivalentes. Isso porque apesar dos sistemas analisados fazerem parte de um mesmo domínio, eles não foram necessariamente desenvolvidos por uma mesma equipe, em um mesmo período de tempo ou dentro de um mesmo departamento. Abordagens de comparação de modelos para a detecção de diferenças costumam considerar diferentes versões de um mesmo modelo na comparação, onde os elementos equivalentes possuem o mesmo nome ou identificador (ex: (CHEN *et al.*, 2003) (MEHRA *et al.*, 2005) (OLIVEIRA, 2005)), sendo essas informações utilizadas como base para estabelecer a comparação. Como essas premissas não podem ser assumidas na comparação de modelos de diferentes sistemas legados em um domínio, a comparação se torna uma tarefa mais árdua, devendo levar em conta a grande variedade de elementos equivalentes com nomes e estruturas distintos encontrados nos diferentes sistemas analisados. Além disso, ArchToDSSA visa a detecção das variabilidades e opcionalidades do domínio, o que também não costuma ser o foco das abordagens existentes de comparação de modelos.

Partindo desta Introdução, o restante do artigo está organizado da seguinte forma: a Seção 2 discute trabalhos relacionados em ED e LP, descrevendo seu apoio à especificação de arquiteturas de referência, além de trabalhos de comparação de modelos; a Seção 3 apresenta a abordagem ArchToDSSA, descrevendo seu processo constituído de 3 fases, a saber: detecção de opcionalidades, detecção de variabilidades e criação de DSSA; a Seção 4 mostra um exemplo prático de utilização da abordagem e o seu apoio ferramental; e a Seção 5 apresenta conclusões e trabalhos futuros.

2. Trabalhos Relacionados

A maior parte das abordagens de ED e LP existentes costuma oferecer apoio à especificação de arquiteturas de referência de domínio no sentido da engenharia progressiva, i.e. partindo da análise para o projeto (ex: FORM (KANG *et al.*, 2002), CBD-Arch-DE (BLOIS, 2006), KobrA (ATKINSON *et al.*, 2002) e PLUS (GOMAA, 2004)). Assim, não contemplam o apoio à análise dos sistemas legados no processo, embora argumentem que esses representem uma das fontes de informação essenciais para a análise de domínio. Nesse contexto, ArchToDSSA oferece apoio à comparação das arquiteturas recuperadas de sistemas legados no domínio para a detecção das suas opcionalidades e variabilidades e, conseqüentemente, criação da DSSA.

As abordagens de ED e LP que focam na engenharia reversa, em geral, não oferecem um apoio sistemático à extração e análise de modelos dos sistemas legados para apoiar a criação de arquiteturas de referência de domínio, estando focadas em algum aspecto específico, como a extração ou avaliação da adequação de componentes à arquitetura de referência (GANESAN & KNODEL, 2005; KNODEL & MUTHIG, 2005) ou detecção e refatoração de variabilidades no código (ALVES *et al.*, 2007). Em contrapartida, ArchToDSSA provê técnicas e critérios para a detecção de

opcionalidades, variabilidades e criação/exportação da DSSA, complementando, no contexto da abordagem LegaToDSSA (VASCONCELOS, 2007), um processo sistemático de apoio à extração e análise de modelos de sistemas legados no domínio.

Finalmente, as abordagens de comparação de modelos para a detecção de diferenças costumam considerar diferentes versões de um mesmo modelo na comparação, onde os elementos equivalentes devem possuir o mesmo nome, tipo ou identificador (ex: (CHEN *et al.*, 2003) (MEHRA *et al.*, 2005) (OLIVEIRA, 2005)). Dessa forma, não contemplam arquiteturas de diferentes sistemas, onde elementos equivalentes podem possuir nomes e estruturas distintos. ArchToDSSA, por outro lado, incorpora um dicionário de sinônimos que permite que elementos com nomes distintos mas semântica igual sejam considerados equivalentes. Além disso, possibilita a comparação de nomes por *substrings* comuns e permite considerar ou não o tipo e a estrutura sintática dos elementos durante a comparação.

3. ArchToDSSA: Comparação de Modelos Arquiteturais de Sistemas Legados para Apoiar a Criação de Arquiteturas de Referência de Domínio

ArchToDSSA visa a comparação de modelos arquiteturais recuperados de sistemas legados para apoiar a criação de arquiteturas de referência de domínio, que podem ser utilizadas no contexto da ED e da LP, e a partir da qual novas aplicações no domínio podem ser instanciadas. A abordagem pode ser adotada, por exemplo, por empresas que possuam uma base de sistemas legados e estejam pretendendo migrar para abordagens de ED e de LP.

A fim de identificar elementos opcionais ou mandatórios e variantes, invariantes ou VPs, gerando arquiteturas de referência, a abordagem ArchToDSSA propõe um processo de 3 fases, conforme mostra a Figura 1. O processo envolve a Detecção de Opcionalidades, Detecção de Variabilidades e Criação da DSSA. O papel responsável pela execução das atividades é o Engenheiro de Domínio ou o engenheiro em uma abordagem de LP, responsável pela elaboração da arquitetura de referência de domínio. Vale ressaltar que ArchToDSSA gera arquiteturas de referência parcialmente especificadas, podendo gerar diferentes arquiteturas a partir de um mesmo conjunto de arquiteturas legadas, sendo gerada, entretanto, uma arquitetura por vez. As DSSAs devem ter a sua especificação complementada pelo engenheiro que apóia o processo.

O processo apresentado na Figura 1 segue a notação do metamodelo SPEM (*Software Process Engineering Metamodel*) (OMG, 2005), definido pela OMG, que faz uso de uma notação estendida de diversos modelos da UML com o intuito de permitir a modelagem de processos. Como pode ser visto na Figura 1, que representa uma extensão do diagrama de atividades da UML, uma vez executado um processo de engenharia reversa, como, por exemplo, ArchMine (VASCONCELOS, 2007), que gere um conjunto de Arquiteturas Recuperadas de Sistemas no Domínio, a atividade de Detecção de Opcionalidades pode ser iniciada. Ela analisa as arquiteturas recuperadas e gera como saída uma lista de Elementos Equivalentes no domínio, que representam elementos semanticamente correspondentes nas diferentes arquiteturas analisadas, e as Arquiteturas Recuperadas com as suas Opcionalidades. Com base nesses artefatos gerados pela primeira atividade do processo, a atividade de Detecção de Variabilidades detecta os pontos de variação e variantes do domínio, gerando Arquiteturas Recuperadas

com Opcionalidades, VPs e Variantes. A Criação da DSSA utiliza as informações geradas ao longo do processo para gerar Arquiteturas de Referência de Domínio parcialmente especificadas.

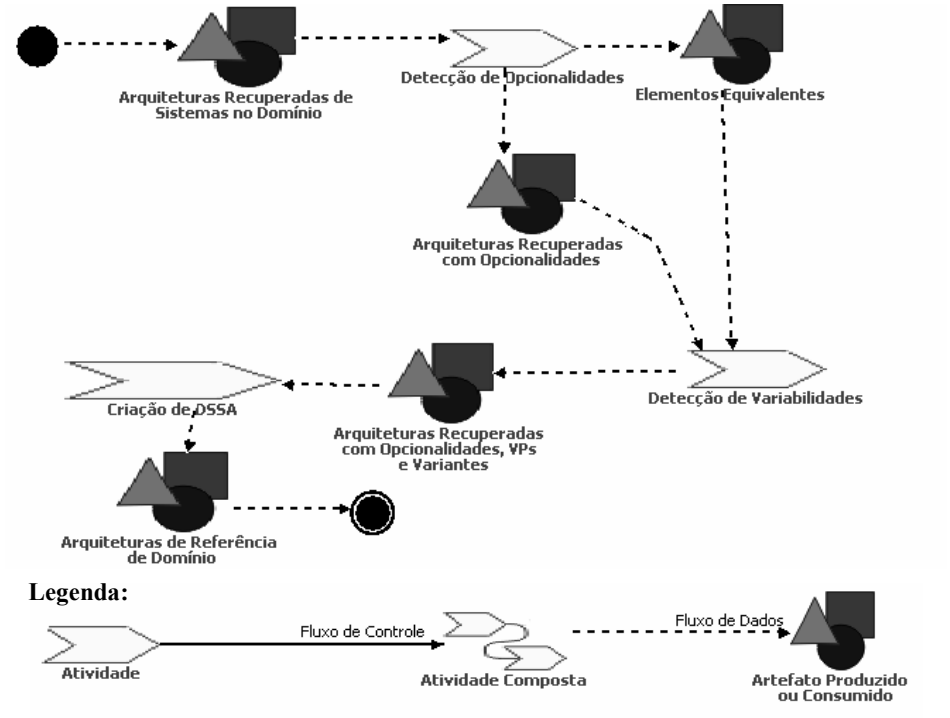

Figura 1. Processo proposto da abordagem ArchToDSSA

ArchToDSSA assume como pré-requisitos que os sistemas sejam Orientados a Objetos (OO) e que as suas arquiteturas estejam representadas em UML, descritas em formato XMI. Assim, os elementos arquiteturais são representados através de pacotes e a sua estrutura interna através de classes, com os seus relacionamentos. As subseções a seguir descrevem detalhadamente as fases ou atividades do processo de ArchToDSSA.

3.1. Detecção de Opcionalidades

Nesta primeira fase da abordagem, o objetivo é identificar dentre as arquiteturas recuperadas dos sistemas legados, as similaridades e diferenças entre seus respectivos elementos arquiteturais. Dessa forma, é possível identificar elementos mandatórios e opcionais no domínio, ou seja, as opcionalidades do domínio em questão. Essas opcionalidades são identificadas tanto em nível de elemento arquitetural quanto em nível de suas classes (estrutura interna).

Para a identificação de elementos mandatórios e opcionais, o primeiro passo é a identificação de equivalências (*matches*). Equivalências representam elos de ligação entre elementos semanticamente correspondentes de diferentes arquiteturas. Elementos equivalentes podem apresentar o mesmo nome ou nomes diferentes nas arquiteturas comparadas em função delas se originarem de diferentes aplicações do domínio. A Figura 2 apresenta um exemplo considerando um domínio escolar, onde "Aluno" e "Estudante", por exemplo, representam o mesmo conceito, embora apresentando nomes

diferentes.

Essa equivalência semântica em ArchToDSSA é obtida através de um dicionário de sinônimos que é alimentado pelo Engenheiro de Domínio. A cada ligação estabelecida pelo Engenheiro de Domínio entre elementos de diferentes arquiteturas com nomes diferentes, uma entrada no dicionário é criada, definindo esses nomes como sinônimos. O Engenheiro de Domínio pode ainda definir diretamente sinônimos no dicionário sem necessariamente estar efetuando ligações entre as arquiteturas. Assim, a comparação das equivalências com novas arquiteturas no mesmo domínio é simplificada através do dicionário, o qual é constantemente alimentado durante o processo.

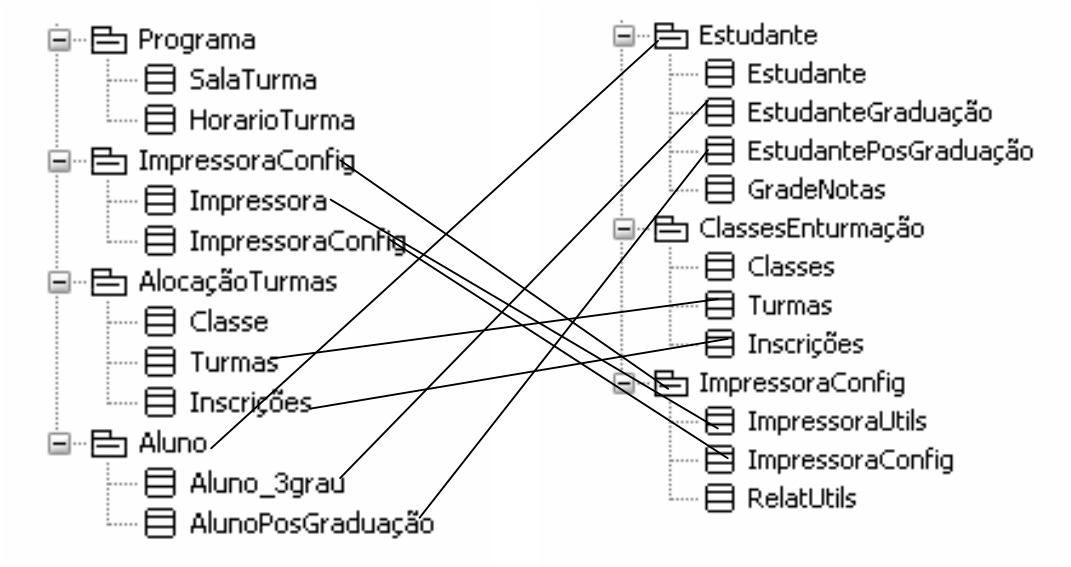

Figura 2. Equivalências entre diferentes arquiteturas de um domínio

Além da comparação de nomes iguais ou sinônimos, a abordagem admite ainda 3 possibilidades de configuração para a determinação de equivalências:

- **Lista de Palavras Ignoradas**: ArchToDSSA permite que palavras sem semântica "forte", como Gerente, Utils, Controlador etc., sejam ignoradas na detecção de equivalências. Essas palavras não contribuem semanticamente com a comparação. Assim, uma lista de palavras ou abreviações que devem ser ignoradas é mantida.
- **Comparação por *Substrings***: é possível que os nomes sejam divididos em partes durante a comparação. Os critérios para a divisão em partes envolvem a detecção de uma letra maiúscula ou um sinal de sublinhado. Dessa forma, no nome "ImpressoraUtils" poderão existir duas partes, i.e. "Impressora" e "Utils", e no nome "AlunoPosGraduação" três partes, i.e. "Aluno", "Pos" e "Graduação". Após dividir os nomes em partes, é verificado para cada parte (em cada nome), se a referida parte se encontra na "lista de palavras a serem ignoradas", caso se encontre, essa parte é retirada da lista de partes da palavra. Em seguida, cada parte do primeiro nome é comparada com cada parte do segundo nome. Se uma parte de um nome possui uma parte igual em outro nome, essas duas partes são marcadas e não podem mais ser envolvidas na comparação das outras partes dos nomes comparados. Assim, por exemplo, as *strings* BaBeBa e BaBaBe são

consideradas iguais, uma vez que as *substrings* "Ba", "Be" e "Ba" são encontradas na segunda *string*. Entretanto, as *strings* BaBeBa e BaBeBe são diferentes uma vez que "Ba" e "Be" são encontradas na segunda *string*, mas "Ba" não é encontrada em uma segunda ocorrência.

- **Comparação de Elementos do Mesmo Tipo**: indica se deve ser considerado o tipo do elemento na comparação, i.e. classes só devem ser comparadas com classes e pacotes só devem ser comparados com pacotes.

Em relação ao exemplo apresentado na Figura 2, observa-se que a comparação é realizada por sinônimo, comparação de *substrings*, palavras ignoradas e tipo do elemento arquitetural. Por exemplo, "Aluno" pacote (i.e. elemento arquitetural) corresponde a "Estudante" pacote, ao passo que "Aluno_3grau" classe (i.e. parte da estrutura interna de um elemento arquitetural) corresponde a "EstudanteGraduação", também classe, nas diferentes arquiteturas. "Graduação" e "3grau" são sinônimos nesse caso, bem como "Aluno" e "Estudante".

Considerando a lista de palavras ignoradas e a comparação por *substrings*, observa-se que a *substring* "Utils" é ignorada nessa comparação e que, dessa forma, "Impressora" na 1ª arquitetura se torna equivalente à "ImpressoraUtils" na 2ª arquitetura. Além disso, considerando ainda a comparação por tipo do elemento, observa-se que "ImpressoraConfig" pacote corresponde a "ImpressoraConfig" pacote e que "ImpressoraConfig" classe corresponde a "ImpressoraConfig" também classe.

Todas essas opções podem ser determinadas como aplicáveis ou não pelo Engenheiro de Domínio, uma vez que a sua utilização em ArchToDSSA é flexível. Nessas comparações entre as arquiteturas, é sempre registrado o número de arquiteturas onde a equivalência é detectada. Quando elementos equivalentes são encontrados em todas as arquiteturas comparadas, eles são candidatos a elementos mandatórios no domínio, ao passo que se os elementos equivalentes são encontrados em algumas das arquiteturas comparadas apenas, eles são candidatos a elementos opcionais do domínio.

3.2. Detecção de Variabilidades

Na segunda fase de ArchToDSSA, as arquiteturas recuperadas e a lista de equivalências obtidas na fase 1 são analisadas para que se possa definir os VPs e as suas respectivas variantes no domínio, ou seja, as variabilidades do domínio em questão.

Conforme mencionado, a abordagem recebe como entrada arquiteturas recuperadas de sistemas legados OO, representadas através de modelos de pacotes e de classes da UML e descritas em formato XMI. Com base nessas informações, é possível descobrir os relacionamentos entre as classes. Os relacionamentos de herança e de implementação, especificamente, são de grande interesse para a descoberta de possíveis VPs e suas respectivas variantes. De acordo com a Odyssey-FEX (OLIVEIRA, 2006), notação adotada por ArchToDSSA para a representação de variabilidades e opcionalidades do domínio, um VP pode ser representado por uma superclasse ou interface no modelo de classes, e suas variantes podem ser representadas por subclasses ou classes que implementam uma interface no modelo de classes.

Além disso, é possível que o Engenheiro de Domínio determine em quantas arquiteturas uma dada interface ou superclasse deve possuir equivalência para que ela possa ser considerada um VP. Dessa forma, ArchToDSSA é capaz de indicar candidatos

a VPs e variantes, sendo que as variantes candidatas são todas as subclasses encontradas nas diferentes arquiteturas onde a superclasse que representa o VP possua equivalência ou todas as classes nas diferentes arquiteturas que implementem a interface que representa o VP.

Entretanto, essas regras de representação de VPs e variantes da Odyssey-FEX, embora tendo sido derivadas com base em estudos empíricos (OLIVEIRA, 2006), envolvendo a análise de modelos de classes de três domínios distintos, podem não ser válidas para todos os casos. Assim, é importante que o Engenheiro de Domínio possa determinar aleatoriamente VPs e variantes nas diferentes arquiteturas comparadas, bem como recusar sugestões de VPs e variantes detectados através do uso da abordagem.

3.3. Criação da DSSA

Finalmente, na última fase de ArchToDSSA, é selecionada uma das arquiteturas recuperadas como base para a criação da arquitetura de referência de domínio. A seleção de uma arquitetura é necessária para evitar a existência de estruturas inconsistentes na arquitetura de referência resultante, o que poderia ocorrer caso elementos de diferentes arquiteturas fossem combinados aleatoriamente para gerar a arquitetura de referência. Dessa forma, assegura-se que todos os elementos e relacionamentos dessa arquitetura "base" estarão incondicionalmente presentes na arquitetura de referência. Vale ressaltar que a arquitetura de referência, assim como as arquiteturas analisadas por ArchToDSSA, é representada em formato XMI. Elementos opcionais de outras arquiteturas, que não a arquitetura "base" selecionada, podem também compor a arquitetura de referência.

Diferentes arquiteturas de referência podem ser criadas para o mesmo conjunto de sistemas legados, com base em diferentes escolhas referentes à arquitetura base e elementos opcionais realizadas pelo Engenheiro de Domínio. A fim de facilitar essas escolhas, ArchToDSSA apresenta em cada arquitetura os elementos opcionais, mandatórios, VPs e variantes. Entretanto, essas escolhas não são apoiadas por ArchToDSSA, podendo ser levada em conta a experiência do Engenheiro do Domínio.

As informações de opcionalidades e variabilidades são combinadas, uma vez que representam propriedades ortogonais na Odyssey-FEX, sendo então estabelecidos VPs mandatórios e opcionais, bem como variantes mandatórias e opcionais, além de invariantes mandatórias e opcionais. Elementos opcionais selecionados de outras arquiteturas que não a arquitetura “base” são marcados como "não definidos" na arquitetura de referência, marcação essa permitida pela notação Odyssey-FEX, o que indica que eles ainda precisarão ter a sua especificação complementada na arquitetura de referência. Esses elementos são propositadamente incorporados de forma "solta" à arquitetura de referência, uma vez que não há como garantir que seus relacionamentos na sua arquitetura original se mantenham na arquitetura de referência, pois muitos dos elementos com os quais eles se relacionam direta ou indiretamente, podem não possuir equivalência na arquitetura "base". Ainda que possuam equivalência, não há como assegurar que as suas estruturas internas sejam exatamente iguais nas diferentes arquiteturas para garantir que as ligações continuem fazendo sentido.

A exceção se encontra para o caso das variantes opcionais incorporadas de outras arquiteturas, que são ligadas na arquitetura de referência através de herança ou

implementação de interface. Mas, ainda assim, elas são marcadas como "não definidas", pois um exame minucioso da sua compatibilidade estrutural e semântica em relação ao VP ainda deve ser realizado pelo Engenheiro de Domínio. Convém ressaltar, ainda, que caso sejam selecionados elementos equivalentes opcionais de diferentes arquiteturas para a criação da arquitetura de referência, ArchToDSSA admite apenas uma ocorrência do elemento na arquitetura final. Se ele existir na arquitetura "base" escolhida, será então priorizada a sua ocorrência nessa arquitetura.

Os elementos selecionados de outras arquiteturas, que não a arquitetura "base", estarão hierarquicamente subordinados ao seu elemento superior mais direto com equivalência na arquitetura "base" (i.e. um subpacote ou um pacote). Caso essa equivalência não seja encontrada em nenhum nível da hierarquia para o elemento incorporado de outra arquitetura, ele é alocado diretamente à raiz da arquitetura de referência, ou seja, a um pacote "DSSAmodel" criado no XMI.

Conforme mencionado, as arquiteturas de referência resultantes são representadas através de XMI. Uma vez que opcionalidades e variabilidades, bem como o conceito de "não definido", não fazem parte do metamodelo da UML, essas propriedades são representadas através de um mecanismo de extensão da UML, i.e. valores chaveados (*tagged values*). Valores chaveados permitem definir novos tipos de propriedades, adicionando tuplas de <nome, valor> a qualquer elemento de modelo. Dessa forma, cada elemento na arquitetura de referência resultante conterá 3 tuplas de <nome, valor>, a saber:

- Nome = "opcionalidade"; valor = "mandatório" ou "opcional";
- Nome = "variabilidade"; valor = "VP", "variante" ou "invariante";
- Nome = "não definido"; valor = "verdadeiro" ou "falso".

Os elementos "não definidos" na arquitetura de referência gerada a caracterizam como parcialmente especificada, devendo ser analisada e complementada pelo Engenheiro de Domínio antes de ser instanciada para novas aplicações.

4. Exemplo de Utilização da Abordagem e seu Apoio Ferramental

A fim de ilustrar a utilização da abordagem ArchToDSSA, um exemplo hipotético envolvendo um conjunto de 3 arquiteturas no domínio de telefonia móvel é apresentado nesta seção. Em (STOERMER & O'BRIEN, 2001), os autores comentam que um número mínimo de 3 a 4 arquiteturas de sistemas no domínio deve ser analisado para a detecção de opcionalidades e variabilidades do domínio.

A abordagem ArchToDSSA é executada com o apoio da ferramenta ArchToDSSATool, a qual pode ser utilizada de forma isolada ou integrada ao ambiente de reutilização Odyssey. Em ambos os casos, ela lê e gera arquivos em formato XMI, descrevendo modelos arquiteturais em UML. Dessa forma, embora possa ser utilizada de forma integrada ao Odyssey, ArchToDSSATool apresenta uma certa independência de ambiente de desenvolvimento.

A Figura 3 apresenta a tela principal da ArchToDSSATool, contemplando a primeira fase da abordagem. As 3 arquiteturas do domínio de telefonia móvel foram carregadas na ferramenta. ArchToDSSATool permite selecionar a arquitetura desejada em cada coluna, como pode ser visto pela seleção da XMICelular1 na primeira coluna,

apresentando as arquiteturas lado a lado, de duas em duas. A ferramenta trabalha com o conceito de *working set* (conjunto de trabalho), podendo ser definido um *working set* para cada domínio, contendo as arquiteturas dos sistemas no domínio, o dicionário e as configurações estabelecidas para a criação da DSSA.

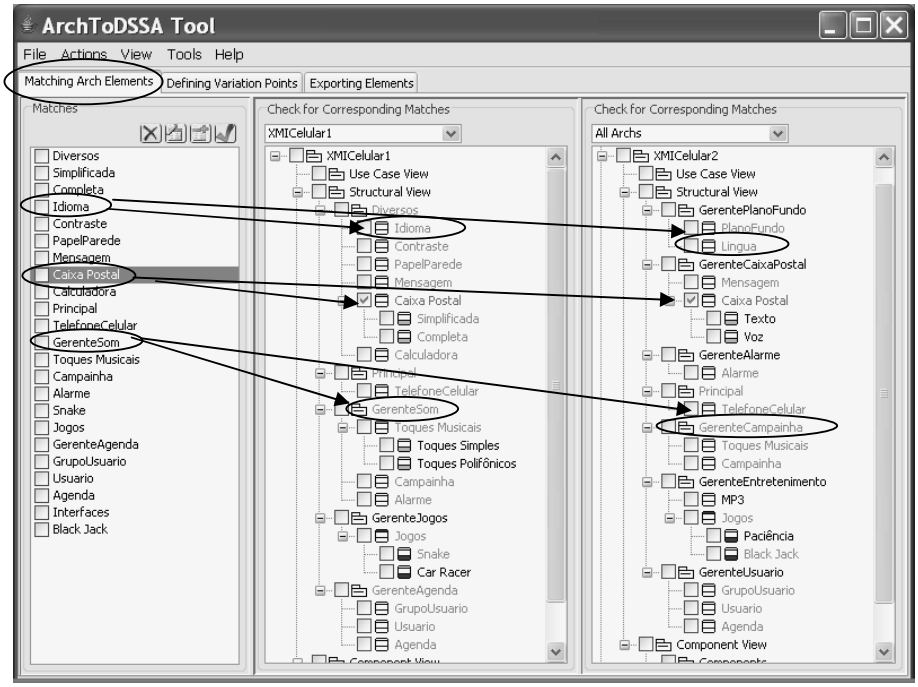

Figura 3. Tela principal da ferramenta ArchToDSSATool

A Figura 4 apresenta a tela de configuração de parâmetros para a realização da primeira fase da abordagem, ou seja, a detecção de equivalências (*matches*) e, conseqüentemente, de opcionalidades do domínio. Através desta tela, o usuário pode indicar: se deseja que o dicionário seja automaticamente alimentado quando valida ou cria uma equivalência (*match*) – opção: *Auto Feed Dictionary when Validating Match;* se o dicionário deve ser utilizado para sugerir equivalências – opção: *Use Dictionary when Suggesting Matches*; se a comparação por *substrings* deve ser adotada – opção: *Use Substring Comparison when Suggesting Matches*; e se a comparação deve levar em conta o tipo do elemento, ou seja, classes só devem ser comparadas com classes e pacotes com pacotes – opção: *Compare only Similar Elements when Suggesting Matches*. Além disso, o usuário pode ainda indicar em quantas arquiteturas a equivalência deve ser encontrada para que ela seja sugerida como um *match*.

Uma vez que todas as opções apresentadas na Figura 4 tenham sido selecionadas e o dicionário do domínio alimentado, a ferramenta ArchToDSSATool pode gerar, de forma automática, uma lista de equivalências entre as arquiteturas, como mostra a coluna da esquerda da tela principal da ferramenta na Figura 3. Neste caso, a equivalência ressaltada, i.e. Caixa Postal, possui correspondência em todas as arquiteturas, representando um elemento mandatório. A equivalência Idioma leva em conta ocorrências das palavras Idioma e Língua nas diferentes arquiteturas, uma vez que esta correspondência foi alimentada no dicionário de sinônimos. A equivalência

GerenteSom inclui ocorrências de GerenteSom e GerenteCampainha (Figura 3), uma vez que as palavras Som e Campainha também representam sinônimos no dicionário.

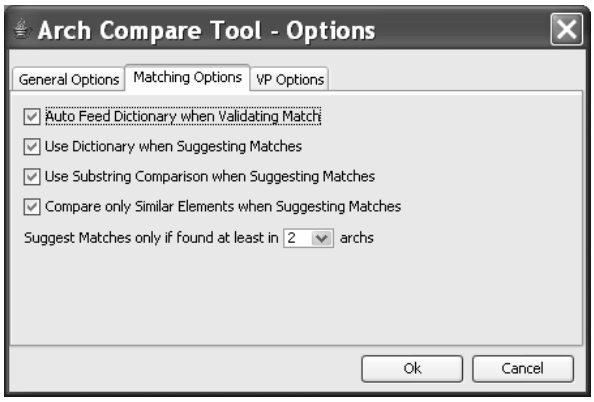

Figura 4. Tela de configuração de parâmetros da ArchToDSSATool

O Engenheiro de Domínio deve validar as equivalências que julgar corretas, tendo ainda a possibilidade de criar outras equivalências manualmente. Dessa forma, ele pode passar à fase 2 da abordagem, i.e. Detecção de Variabilidades, como mostra a Figura 5. Neste exemplo, o ponto de variação Jogos, que representa uma interface no modelo de classes, está selecionado, possuindo as variantes *Snake, Car Racer*, Paciência e *Black Jack*. Além desse VP, a ferramenta também detectou automaticamente o VP Caixa Postal, com as variantes Simplificada, Completa, Texto e Voz, e o VP Toques Musicais (Figura 5). Esses elementos representam hierarquias nas arquiteturas comparadas, ou seja, uma estrutura de herança que é uma candidata a VP.

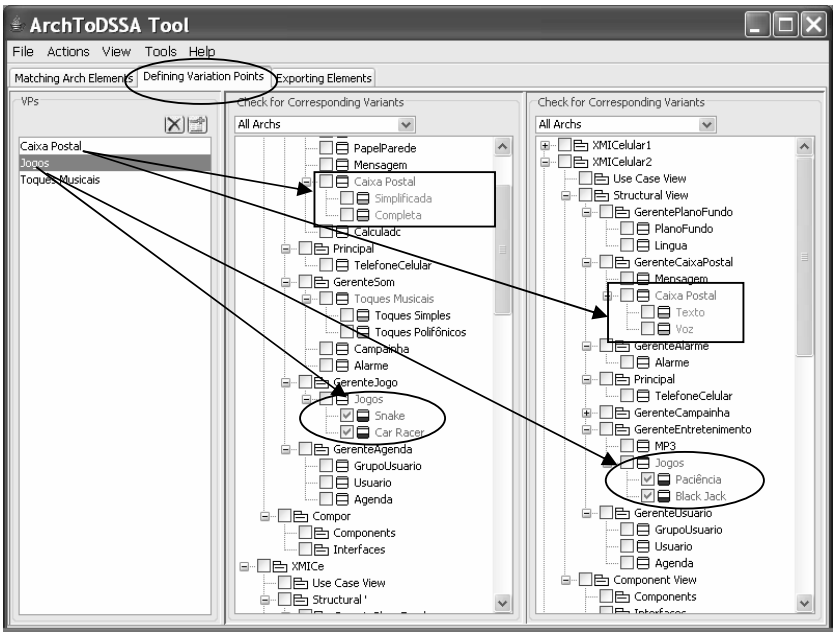

Figura 5. Detecção de variabilidades na ArchToDSSATool

A fase seguinte é a fase de criação e exportação da DSSA, para o ambiente Odyssey ou para o formato XMI, contendo as opcionalidades e variabilidades definidas. A Figura 6 apresenta a DSSA criada para o domínio de telefonia móvel no ambiente de

reutilização Odyssey. Os pacotes e classes ficam na *Structural View* do domínio no ambiente (lado esquerdo da Figura 6), podendo ser visualizados através de diagramas. Vale ressaltar que no ambiente Odyssey o modelo arquitetural gerado pode ser mapeado para o modelo de características do domínio, através de regras de mapeamento propostas em (OLIVEIRA, 2006). Dessa forma, é possível gerar um modelo de termos do domínio, que reflita a sua semântica e apóie a Análise do Domínio.

Como pode ser observado, Jogos é um VP (Figura 6), e como a arquitetura XMICelular1 foi a arquitetura-base selecionada para a criação da DSSA, as variantes Paciência e Black Jack ficam como não definidas (i.e. not defined), uma vez que constam da arquitetura XMICelular2 (Figura 5). Além disso, todas as variantes são opcionais, como pode ser visto pelo estereótipo Optional. Jogos, entretanto, é um VP mandatório. A classe BlueTooth também representa um elemento opcional do domínio. Vale ressaltar que no Odyssey as propriedades não definido, opcionalidade e variabilidade são visualmente apresentadas como estereótipos.

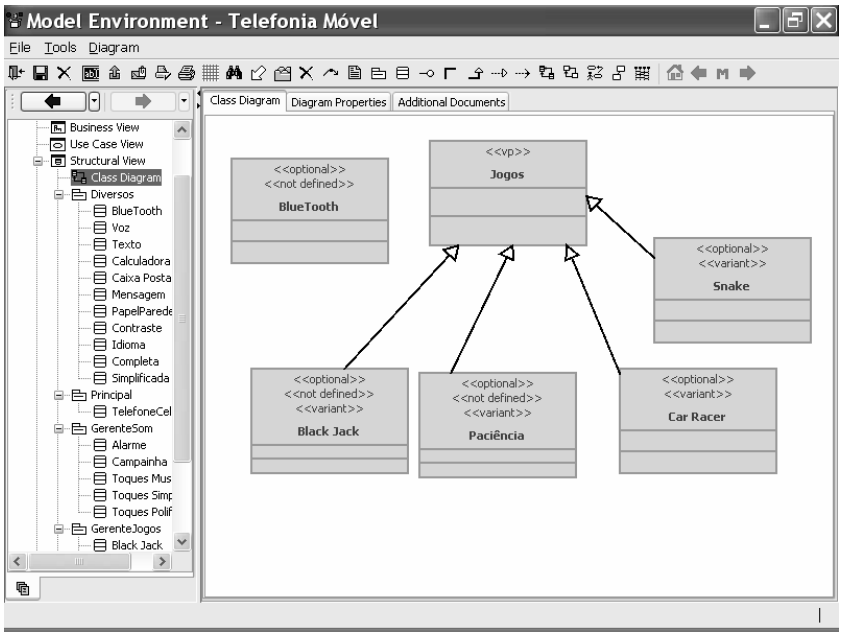

Figura 6. DSSA para o domínio de telefonia móvel no ambiente Odyssey

5. Conclusões e Trabalhos Futuros

ArchToDSSA representa uma abordagem de comparação de modelos arquiteturais de sistemas legados para apoiar a criação de arquiteturas de referência de domínio, que podem ser utilizadas no contexto da ED e LP. A abordagem apóia a geração de arquiteturas de referência de domínio parcialmente especificadas, permitindo a geração de diferentes DSSAs para o mesmo conjunto de arquiteturas legadas. Dessa forma, além de representar uma abordagem de comparação e extração de diferenças entre modelos, como outras abordagens existentes (CHEN *et al.*, 2003; MEHRA *et al.*, 2005; OLIVEIRA, 2005), ArchToDSSA contribui para a área de reutilização de software, identificando opcionalidades e variabilidades estruturais no domínio. Embora ArchToDSSA seja voltada à análise de sistemas legados, vale ressaltar que a abordagem pode ser utilizada também para a comparação de modelos arquiteturais

desenvolvidos através da engenharia progressiva, desde que esses atendam aos pré-requisitos mencionados para ArchToDSSA, ou seja, serem OO e possuírem as arquiteturas descritas em UML/XMI. ArchToDSSA é atualmente limitada à análise de sistemas com essas características.

As opcionalidades e variabilidades são identificadas tanto em nível de elemento arquitetural quanto em nível de suas classes, i.e. estrutura interna. Entretanto, nesse momento, a opcionalidade das classes não impacta a opcionalidade do elemento arquitetural, o que pode ser visto como uma necessidade para trabalho futuro. Além disso, seria interessante detectar diferenças em um nível de granularidade mais fino, ou seja, em nível de métodos e atributos, pois opcionalidades e variabilidades em um domínio podem se manifestar nesse nível "fino" de granularidade (MORISIO *et al.*, 2000). Seria interessante detectar também variabilidades e opcionalidades em outros modelos de domínio, como os modelos comportamentais, além de detectar variações também em termos dos requisitos não-funcionais do domínio.

Finalmente, ArchToDSSA e o seu apoio ferramental foram avaliados através de dois estudos de observação (KÜMMEL, 2007), um estudo envolvendo estudantes e o outro envolvendo profissionais da indústria, tendo sido analisados modelos arquiteturais hipotéticos. Esses estudos permitiram obter *feedback* e indícios sobre a eficiência de ArchToDSSA no apoio à criação de DSSAs.

Referências

ALVES, V., MATOS JR, P., COLE, L.*, et al.*, 2007, "Extracting and Evolving Code in Product Lines with Aspect-Oriented Programming", *Transactions on Aspect-Oriented Software Development (TAOSD): Special Issue on Software Evolution, To Appear*.

ATKINSON, C., BAYER, J., BUNSE, C.*, et al.*, 2002, *Component-based Product Line Engineering with UML*, Boston, Addison-Wesley Longman Publishing Co., Inc.

BLOIS, A.P.B., 2006, *Uma Abordagem de Projeto Arquitetural Baseado em Componentes no Contexto de Engenharia de Domínio*, Tese de D.Sc., COPPE, UFRJ, Rio de Janeiro, Brasil.

CHEN, P., CRITCHLOW, M., CARG, A.*, et al.*, 2003, "Differencing and Merging within an Evolving Product Line Architecture". In: *International Workshop on Software Product-Family Engineering*, pp. 269-281, Siena, Italy, November.

GANESAN, D., KNODEL, J., 2005, "Identifying Domain-Specific Reusable Components from Existing OO Systems to Support Product Line Migration". In: *First International Workshop on Reenegineering Towards Product Lines (R2PL),* pp. 16-20, Pittsburgh, PA, USA, November.

GOMAA, H., 2004, *Designing Software Product Lines with UML: from Use Cases to Pattern-Based Software Architectures*, Addison-Wesley Professional.

KANG, K.C., LEE, J., DONOHOE, P., 2002, "Feature-Oriented Product Line Engineering", *IEEE Software*, v. 9, n. 4 (Jul./Aug 2002), pp. 58-65.

KNODEL, J., MUTHIG, D., 2005, "Analyzing the Product Line Adequacy of Existing Components". In: *First Workshop on Reengineering towards Product Line*

(R2PL), pp. 21-25, Pittsburgh, PA, USA, November.

KÜMMEL, G., 2007, *Uma Abordagem para a Criação de Arquiteturas de Referência de Domínio a partir da Comparação de Modelos Arquiteturais de Aplicações*, Dissertação de M.Sc., COPPE, UFRJ, Rio de Janeiro, Brasil.

LEE, K., KANG, K.C., LEE, J., 2002, "Concepts and Guidelines of Feature Modeling for Product Line Software Engineering". In: *Software Reuse: Methods, Techniques, and Tools : 7th International Conference, ICSR-7, Proceedings* pp. 62 - 77, Austin, TX, USA, April, 2002.

MEHRA, A., GRUNDY, J., HOSKING, J., 2005, "A Generic Approach to Supporting Diagram Differencing and Merging for Collaborative Design". In: *20th IEEE/ACM International Conference on Automated Software Engineering (ASE'05)*, pp. 204-213, Long Beach, CA, USA, November.

MORISIO, M., TRAVASSOS, G.H., STARK, M.E., 2000, "Extending UML to Support Domain Analysis". In: *Proceedings of the The Fifteenth IEEE International Conference on Automated Software Engineering (ASE'00)*, pp. 321-324, Grenoble, France, September.

ODYSSEY, 2007, "Odyssey: Infra-Estrutura de Reutilização baseada em Modelos de Domínio". In: http://reuse.cos.ufrj.br/odyssey, accessed in 03/05/2007.

OLIVEIRA, H.L.R., 2005, *Odyssey-VCS: Uma Abordagem de Controle de Versões para Elementos da UML*, Dissertação de M.Sc., COPPE, UFRJ, Rio de Janeiro, RJ, Brasil.

OLIVEIRA, R., 2006, *Formalização e Verificação de Consistência na Representação de Variabilidades*, Dissertação de M.Sc., COPPE, UFRJ, Rio de Janeiro, Brasil.

OMG, 2005, *Software Process Engineering Metamodel (SPEM) Specification, version 1.1*, formal/05-01-06, Object Management Group.

PRIETO-DIAZ, R., ARANGO, G., 1991, "Domain Analysis Concepts and Research Directions", *PRIETO-DIAZ, R., ARANGO, G. (eds), Domain Analysis and Software Systems Modeling, IEEE Computer Society Press*, pp. 9-33.

STOERMER, C., O'BRIEN, L., 2001, "MAP: Mining Architectures for Product Line Evaluations". In: *3rd Working IFIP Conference on Software Architecture (WICSA)*, pp. 35-44, Amsterdam, Holland, August.

SUGUMARAN, V., PARK, S., KANG, K.C., 2006, "Software Product Line Engineering", *Communications of the ACM*, v. 49, n. 12.

VASCONCELOS, A., 2007, *Uma Abordagem de Apoio à Criação de Arquiteturas de Referência de Domínio baseada na Análise de Sistemas Legados*, Tese de D.Sc., PESC - COPPE, UFRJ, Rio de Janeiro, Brasil.

XAVIER, J.R., 2001, *Criação e Instanciação de Arquiteturas de Software Específicas de Domínio no Contexto de uma Infra-Estrutura de Reutilização*, Dissertação de M.Sc., COPPE, UFRJ, Rio de Janeiro, Brasil.