UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIÊNCIAS EXATAS
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Iuri Malinoski Teixeira

Síntese Automática de Interfaces Gráficas de Usuário para Sistemas de Informação em Saúde

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Orientador: Regina Maria Maciel Braga Villela
Coorientador: Antônio Tadeu Azevedo Gomes

Juiz de Fora
2013

Iuri Malinoski Teixeira

Síntese Automática de Interfaces Gráficas de Usuário para Sistemas de Informação em Saúde

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Aprovada em 26 de Fevereiro de 2013.

BANCA EXAMINADORA

Profa. Dra. Regina Maria Maciel Braga Villela - Orientador
Universidade Federal de Juiz de Fora

Prof. Dr. Antônio Tadeu Azevedo Gomes - Coorientador
Laboratório Nacional de Computação Científica

Prof. Dra. Fernanda Claudia Alves Campos
Universidade Federal de Juiz de Fora

Prof. Dr. Antonio Francisco do Prado
Universidade Federal de São Carlos

Ao amor da minha vida, Adriana Rodrigues do Carmo, amiga e companheira de todas as horas, que apoiou na jornada desta dissertação em todos os momentos bons e, principalmente, os difíceis.

AGRADECIMENTOS

A todos os meus parentes, em particular a minha mãe Marina Teixeira Bandeira, além da equipe MARTIN do LNCC, pelo encorajamento e apoio.

Aos professores Tadeu (LNCC) e Regina (UFJF) pela orientação, amizade e principalmente, pela paciência e tolerância, sem a qual este trabalho não se realizaria.

Aos professores do Departamento de Ciência da Computação da UFJF pelos seus ensinamentos e aos funcionários do curso, que durante esses anos, contribuíram de algum modo para o nosso enriquecimento pessoal e profissional.

"Algo só é impossível até que
alguém duvide e acabe provando
o contrário".
Albert Einstein

RESUMO

A modelagem de dados clínicos para Sistemas de Informação em Saúde (SIS) demanda expertise de domínio. Técnicas de Desenvolvimento Dirigido por Modelos (DDM) permitem uma melhor articulação entre especialistas de domínio e desenvolvedores de SISs e possibilitam reduzir o custo de desenvolvimento desses sistemas. Modelos de dados clínicos baseados em especificações padronizadas e abertas como a do *open*EHR facilitam sobremaneira a aplicação de técnicas de DDM para SISs. Contudo, o uso de modelos de dados clínicos não resolve sozinho o problema fundamental do alto custo de desenvolvimento de SISs. Uma das causas desse problema é a falta de informações arquiteturais nos modelos de dados clínicos. Sem essas informações arquiteturais, o custo de desenvolvimento é deslocado para a especificação das regras de transformação de modelos de dados clínicos em código de SIS (regras estas fundamentais nas técnicas de DDM), uma vez que cada novo SIS a ser gerado implica na especificação de um novo conjunto de regras). Neste contexto, este trabalho apresenta uma estratégia para geração de código de SISs baseada na combinação entre modelos de dados clínicos e informações arquiteturais. Nessa estratégia, o desenvolvedor é capaz de categorizar SISs em diferentes famílias e definir um conjunto de regras de transformação comum a todos os SISs de uma família. Cada família é definida por um conjunto de SISs com estruturas arquiteturais semelhantes e modelos de dados clínicos distintos. O resultado esperado dessa estratégia é um melhor reuso das regras de transformação de modelos. Essa estratégia é empregada para se alcançar o objetivo principal deste trabalho, que é a concepção de um sistema de transformação para a síntese automática de Interfaces Gráficas de Usuário (GUI - Graphic User Interface) para SISs, considerando as especificações *open*EHR e algumas construções presentes em Linguagens de Descrição Arquitetural (ADL), como Acme. Como prova de conceito, esse framework é aplicado em algumas famílias de SIS.

Palavras-chave: Sistema de Informação em Saúde, Modelos de Dados Clínicos, Desenvolvimento Dirigido por Modelos, Transformações de Modelos..

ABSTRACT

The modeling of clinical data for Health Information Systems (HIS) requires domain expertise. Model-Driven Development (MDD) techniques provide a better articulation between domain experts and developers of HISes and enable the reduction in the development cost of these systems. Clinical data models based on *open* standard specifications such as the *open*EHR facilitates the application of MDD techniques for HISes. Nevertheless, the use of clinical data models alone does not solve the fundamental problem of high development cost for HISes. One cause for this problem is the lack of architectural information in clinical data models. Without such architectural information, the development cost is shifted to the specification of transformation rules from clinical data models to HIS code (these rules are fundamental in MDD techniques), since each new HIS to be generated involves the specification of a new set of rules. In this context, this work presents a strategy for code generation of HISes that combines clinical data models and architectural information. In this strategy, the developer is able to categorize HISes in distinct families and define a set of transformation rules that are common to all HISes in a family. Each family is defined by a set of systems with similar architectural structures and distinct clinical data models. The expected result of such a strategy is a better reuse of model transformation rules. This strategy is employed to achieve the main objective of this work, which is to design a transformation system for the automatic synthesis of graphical user interfaces (GUI) for HISes, considering *open*EHR specifications and some constructs present in architectural description languages (ADLs), such as Acme. As a proof of concept, this framework is applied to some HIS families.

Keywords: Health Information System, Clinical Data Models, Model-Driven Software Development, Model Transformation..

LISTA DE FIGURAS

LISTA DE SÍMBOLOS

ADL	Linguagem de Descrição de Arquitetura (*Architecture Description Language*)
AM	*Archetype Model*
ATL	*Atlas Transformation Language*
AToMS	*Acute Myocardial Infarction Teleconsultation and Monitoring System*
CKM	*Clinical Knowledge Manager*
CSS	*Cascading Style Sheets*
DDM	Desenvolvimento Dirigido por Modelos
EMF	*Eclipse Modeling Framework*
GUI	Interface Gráfica de Usuário (*Graphical User Interface*)
HL7	*Health Level 7*
HTML	*HyperText Markup Language*
IAM	Infarto Agudo do Miocárdio
IMT	*Interface Model Transformation*
JSP	*JavaServer Pages*
M2C	Modelo para Código (*Model to Code*)
M2M	Modelo para Modelo (*Model to Model*)
MDA	*Model Driven Architecture*
MLHIM	*Multilevel Healthcare Information Modeling*
MOF	*Meta Object Facility*
OMG	*Object Management Group*
openEHR	*open Electronic Health Record*
PIM	Modelo Independente de Plataforma (*Platform Independent Model*)
QVT	*Query View Transformation*
RES	Registro Eletrônico de Saúde
RM	*Reference Model*
SIS	Sistema de Informação em Saúde
SPL	Linha de Produto de Software (*Software Product Lines*)

SPLiCE	*Software-Product Lines in HealthCarE*
TIC	Tecnologia da Informação e Comunicação
UML	*Unified Modeling Language*

SUMÁRIO

1 INTRODUÇÃO

Tem-se evidenciado ao longo dos anos um uso crescente de recursos computacionais na área da saúde (HAUX, 2010). Parte desse uso é motivada em grande medida pela possibilidade de aumento, propiciado pelas Tecnologias de Informação e Comunicação (TICs) envolvidas, na capacidade e qualidade da coleta, processamento e disseminação de informações médicas relacionadas ao cuidado com a saúde. Neste contexto, (CANTRILL, 2010) reflete sobre o futuro promissor dos Sistemas de Informação em Saúde (SISs), ressaltando os seguintes benefícios potenciais para a área: (a) melhor comunicação entre diferentes provedores de cuidado em saúde (hospitais, clínicas, laboratórios etc.); (b) eliminação de estudos clínicos desnecessários; (c) maior legibilidade em prontuários clínicos; e, principalmente, (d) menos erros médicos. Exemplos concretos de alguns desses benefícios podem ser evidenciados em diferentes práticas médicas:

- Emergência pré-hospitalar (SASSER et al., 2005): as TICs oferecem benefícios no suporte remoto eficaz ao atendimento de pacientes, permitindo que especialistas orientem à distância equipes ambulatoriais no ato do primeiro atendimento em uma situação de emergência médica (como em uma unidade móvel - ambulância - ou de pronto atendimento geograficamente isolada). A tomada de decisão rápida e correta em situações de emergência pré-hospitalar relacionadas a infartos, por exemplo, permite reduzir consideravelmente a mortalidade e morbidade de pacientes que venham a sofrer dessa enfermidade (CORREA et al., 2011).

- Vigilância epidemiológica (CHEN et al., 2010): as TICs permitem agilizar rotinas como análise e interpretação de dados coletados na saúde pública, sendo possível detectar, alertar e notificar eventos que necessitam de atenção iminente para surtos epidemiológicos (WHO, 2012). A antecipação de ações a partir da detecção de surtos epidêmicos é uma das prioridades na saúde, porém, essa detecção consome muito tempo se feita a partir de diagnósticos de laboratórios somente. Nesse sentido, sistemas de vigilância epidemiológica possuem um potencial significativo para acelerar a detecção de surtos epidêmicos.

A despeito dos exemplos acima, nos últimos anos tem-se observado uma descrença na efetividade dos SISs em prover amplamente os benefícios supracitados (VEST, 2010). Em geral, os fatores inibidores para que especialistas, clinicas ou hospitais adotem SISs recaem em três conjuntos principais - quais sejam, organizacional, ambiental e tecnológico. Este trabalho enfoca os fatores inibidores tecnológicos, que incluem (RAGHUPATHI; UMAR, 2008): (i) alto custo de desenvolvimento; (ii) falta de padronização aberta; e (iii) manutenibilidade problemática. A implementação de Registros Eletrônicos de Saúde (RESs) tem sido proposta como uma forma de aumentar a eficácia do cuidado em saúde, mas as expectativas nessa área ainda não são amplamente atendidas principalmente pelo fato da área de saúde ser mais complexa e dinâmica que outras áreas onde TICs já são extensivamente empregadas há algum tempo, como educação, comércio, indústria e finanças, para citar algumas (MCGINN et al., 2011). Em particular, uma das características fundamentais da área de saúde é sua estrutura organizacional inerentemente hierárquica e descentralizada. Isso se reflete nos diferentes níveis de especificidade da informação médica coletada, armazenada e processada pelos SISs dos diferentes provedores de cuidado em saúde, informação esta que, por outro lado, está usualmente associada a um mesmo indivíduo. Disso decorre uma necessidade natural de interoperabilidade entre os RESs desses diferentes provedores, de modo que os benefícios supracitados possam ser efetivamente alcançados no cuidado da saúde dos pacientes.

Muitas soluções foram propostas na literatura para o problema da interoperabilidade de SISs, grande parte delas definindo terminologias e ontologias para RESs (BLOBEL; PHAROW, 2009). Contudo, o alto custo de implementação e manutenção desses RESs tem inibido sua adoção (NHS, 2011). A abordagem baseada em especificações padronizadas para modelos de dados clínicos - como *open*EHR (BEALE, 2002), HL7 (BEELER, 1998) e MLHIM (CAVALINI; COOK, 2011) - tem sido considerada uma alternativa viável para a concepção de RESs interoperáveis (EICHELBERG et al., 2005). Essas especificações, em particular as que adotam uma abordagem de modelagem multinível como *open*EHR e MLHIM, podem facilitar sobremaneira a aplicação de técnicas de Desenvolvimento Dirigido por Modelos (DDM) (STAHL; VÖLTER, 2006) para a geração de código de SISs, trazendo benefícios potenciais adicionais para o desenvolvimento desses sistemas tais como (RAGHUPATHI; UMAR, 2008): (i) baixo custo e *time-to-market*; (ii) independência de plataforma; e (iii) maior manutenibilidade. Isso se deve ao fato

dessas especificações permitirem estratificar conceitos de tecnologia e conceitos clínicos em diferentes níveis. No caso das especificações *open*EHR, por exemplo, os conceitos de tecnologia são definidos no nível de um modelo de referência, sobre o qual comitês multidisciplinares de tecnologistas e médicos produzem os chamados "arquétipos", que definem tipos de informações clínicas (observação, instrução, ação e avaliação) e restringem os tipos de dados definidos no modelo de referência (BEALE, 2002). Assim, o arquétipo "Pressão arterial", encontrado no repositório *Clinical Knowledge Manager* (CKM, 2012) da fundação *open*EHR, é um arquétipo do tipo observação com elementos clínicos como "Sistólico" e "Diastólico", que definem restrições sobre os valores aceitos pelo tipo de dados mais geral "Quantidade" no modelo de referência. Desse modo, um dos maiores desafios no desenvolvimento de um SIS - a transcrição do conhecimento do médico acerca dos dados clínicos relevantes no SIS (CORREA et al., 2011) - é consideravelmente simplificado.

1.1 MOTIVAÇÃO E OBJETIVOS

Apesar dos benefícios potenciais do emprego de técnicas de DDM na geração de código para SISs, poucas iniciativas recentes na literatura de informática médica têm abordado essa linha de pesquisa (LOPEZ; BLOBEL, 2009; JANAMANCHI et al., 2009). Artigos seminais como (RAGHUPATHI; UMAR, 2008; TUOMAINEN et al., 2007) estabelecem uma base inicial em termos do potencial que DDM pode oferecer para toda uma engenharia sistemática de SISs, mas (HAUX, 2010) aponta o emprego de DDM para SISs como um desafio importante ainda a ser tratado para que se alcance um maior nível de interoperabilidade, manutenibilidade e reuso. Como já foi destacado, modelos de dados clínicos multinível como *open*EHR e MLHIM permitem uma melhor articulação entre especialistas de domínio e desenvolvedores, além de facilitar a geração de código para SISs baseada nestes modelos, mas não resolvem sozinho o problema fundamental do alto custo de desenvolvimento de SIS. Uma das causas desse problema é a falta de informações arquiteturais nos modelos de dados clínicos. Sem essas informações arquiteturais, o custo de desenvolvimento é deslocado para a especificação das regras de transformação de modelos de dados clínicos em código de SISs. Processos de transformação de modelos (CZARNECKI; HELSEN, 2006) são cruciais em DDM e dependem da especificação dessas regras, o que demanda conhecimento especializado de linguagens cujas estruturas (em geral baseadas em conceitos de metamodelagem) não fazem parte da prática comum

de desenvolvimento de software. Isso se reflete em última instância em um custo de desenvolvimento potencialmente mais alto que o não emprego de DDM, uma vez que cada novo SIS a ser gerado implica na especificação de um novo conjunto de regras.

Neste contexto, este trabalho apresenta uma estratégia para geração de código de SISs baseada na combinação entre modelos de dados clínicos multiníveis e informações arquiteturais. Nessa estratégia, o desenvolvedor é capaz de categorizar SISs em diferentes famílias e definir um conjunto de regras de transformação comum a todos os SISs de uma família. Cada família é definida por um conjunto de SISs com estruturas arquiteturais semelhantes. Os seguintes exemplos de famílias serão apresentados nesta dissertação: emergência pré-hospitalar e vigilância sindrômica. O que caracteriza um SIS individual em cada uma dessas famílias é o conjunto de dados clínicos de interesse para esse sistema: por exemplo, a informação de "Tônus muscular" é crucial em obstetrícia de emergência (para o método Apgar) mas não faz sentido no atendimento emergencial a infartados. O resultado esperado da estratégia apresentada é um melhor reuso das regras de transformação de modelos, uma vez que se pode desacoplar as regras responsáveis pela tradução das informações arquiteturais (reusáveis no contexto de uma família) daquelas responsáveis pela tradução do modelo de dados clínicos de interesse (reusáveis na medida em que atuam no nível de modelagem relacionado aos conceitos de tecnologia).

A dificuldade em se transcrever o conhecimento do médico acerca dos dados clínicos relevantes em um SIS, como colocado anteriormente, torna o projeto de Interface Gráfica de Usuário (*Graphical User Interface* - GUI) um elemento particularmente importante no desenvolvimento do SIS uma vez que facilita o diálogo entre especialistas de domínio e desenvolvedores de software. Nesse sentido, a estratégia descrita no parágrafo anterior é empregada para se alcançar o objetivo principal deste trabalho, que é apresentar um sistema de transfomação para a síntese automática de GUI para SISs. Esse sistema de transfomação adota as especificações *open*EHR para a modelagem de dados clínicos e se inspira em algumas construções para definição de estilos arquiteturais oferecidas por Linguagens de Descrição Arquitetural (*Architecture Description Languages* - ADLs) como Acme (GARLAN et al., 1997) para a modelagem arquitetural. O sistema de transfomação oferece a modeladores de SISs um estilo arquitetural base que descreve propriedades fundamentais associadas a GUIs, como "Elemento de visualização" e "Elemento de formulário", por exemplo. Essas propriedades permitem enriquecer componentes definidos

em famílias de SISs específicas - estas também descritas como estilos arquiteturais derivados do estilo arquitetural base - com informações de GUI. As regras de transformação associadas a cada estilo arquitetural que descreve uma família específica, bem como as regras de transformação associadas aos tipos de dados clínicos definidos no modelo de referência do *open*EHR, foram concebidas com base em padrões de projeto consagrados, como *Composite* (GAMMA et al., 1995), sendo armazenadas em um repositório de regras que podem ser reusadas em diferentes aplicações do sistema de transfomação.

Como prova de conceito, o sistema de transfomação apresentado foi aplicado a algumas famílias de SIS. A partir dessa prova de conceito discorre-se neste trabalho sobre a hipótese de que a estratégia proposta, além de simplificar a especificação das regras de transformação de modelos, pode também melhor articular especialistas de domínio e desenvolvedores de SIS. Mais especificamente, a prova de conceito demonstra que se pode separar as regras responsáveis pela tradução das informações arquiteturais daquelas responsáveis pela tradução do modelo de dados clínicos de interesse. A separação das regras de transformação permite maior clareza na especificação da tradução de forma intuitiva, propiciando automatização.

1.2 ORGANIZAÇÃO DO TRABALHO

Este trabalho está organizado em cinco capítulos, além deste capítulo introdutório que apresenta as motivações e objetivos da abordagem proposta. Os capítulos restantes serão apresentados a seguir:

- O Capítulo 2 apresenta conceitos relacionados aos assuntos tratados neste trabalho, como técnicas de DDM, considerando metamodelagem e transformação de modelos, além de modelagem da informação na saúde, considerando padronizações como *open*EHR.

- No Capítulo 3 é apresentado o sistema de transfomação proposto, bem como sua arquitetura, a estratégia de transformação de modelos empregada, a metamodelagem envolvida e o repositório de regras de transformações reutilizáveis construído. Além disso, neste capítulo são apresentados alguns trabalhos relacionados a abordagem proposta.

- O Capítulo 4 emprega o sistema de transfomação proposto no contexto de algumas famílias de SISs, bem como define os metamodelos e modelos de domínio necessários para essa experimentação do sistema de transfomação.

- Finalmente, no Capítulo 5, são apresentadas as conclusões e discutidos possíveis trabalhos futuros.

2 PRESSUPOSTOS TEÓRICOS

Este capítulo apresenta os principais conceitos e técnicas nos quais este trabalho se baseia.

2.1 DESENVOLVIMENTO DIRIGIDO POR MODELOS

Modelos capturam a essência da realidade de forma abstrata com o objetivo de auxiliar no desenvolvimento de soluções adequadas para problemas complexos. Na engenharia de software, modelos podem especificar conceitos essenciais e abstratos, considerando tanto o nível de detalhamento do domínio (capturando as principais características do problema) quanto o nível arquitetural e de código (detalhando a solução). Portanto, modelos permitem não só entender como também solucionar problemas na área de desenvolvimento de software.

O uso de UML (*Unified Modeling Language*) (OMG, 2003) como linguagem para a criação de modelos permite uma melhor comunicação entre especialistas e permite também, com o uso de algumas construções específicas, o processamento por máquinas. Tipicamente, os modelos são especificados em um nível de abstração muito alto e os programadores podem ter problemas na codificação destes modelos, o que pode resultar em inconsistências e redundâncias. Para minimizar estes problemas, técnicas de Desenvolvimento Dirigido por Modelos (DDM) (STAHL; VÖLTER, 2006) podem tornar mais eficaz todo o processo de desenvolvimento de software.

Técnicas de DDM baseiam-se no fato de que mudanças realizadas em um nível mais abstrato de modelagem possam ser propagadas para níveis menos abstratos, como, por exemplo, nível de projeto, mitigando redundâncias e inconsistências no processo geral de desenvolvimento de software.

Dois conceitos centrais de DDM que viabilizam o princípio de especialização de modelos são *metamodelos* e *transformação de modelos*. Esses dois conceitos serão descritos nas subseções a seguir.

2.1.1 METAMODELOS

Modelos podem ser definidos segundo uma estrutura que permite o processamento automatizado dos mesmos. No âmbito das técnicas de DDM, essa estrutura é usualmente definida em metamodelos. Um metamodelo é um modelo que representa os metadados dos elementos de modelagem, e das associações entre esses metadados, que podem ser usados na construção de modelos que instanciam esse metamodelo. Esse processo de construção também vale para o metamodelo em si, ou seja, um metamodelo possui um ou mais metametamodelos correspondentes. Em teoria, esse processo é infinito. Na literatura, existem técnicas mais práticas para seu uso, como as quatro camadas de abstração do padrão MDA (*Model-Driven Architecture*) do OMG (*Object Management Group*) (OMG, 2012), como ilustrado na Figura 2.1: metametamodelo (`M3`), metamodelo (`M2`), modelo (`M1`) e instância executável (`M0`). O padrão MDA define o nível M3 com o MOF (*Meta Object Facility*) (OMG, 2006), um modelo que se auto define. Essa técnica de definição de modelos permite maior reuso na fase de projeto do software, bem como maior interoperabilidade do software resultante, que pode ser garantida por construção.

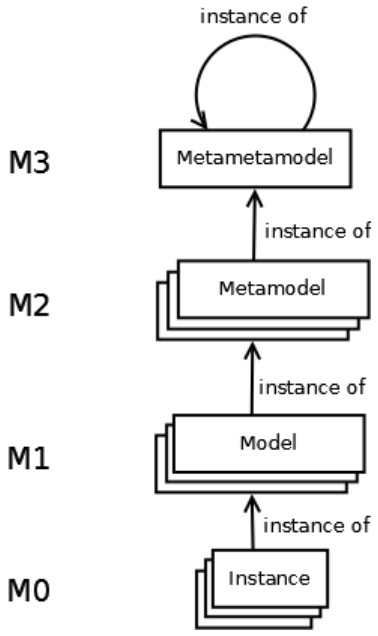


Figura 2.1: Níveis de abstração de modelos (OMG, 2012).

Neste trabalho foi adotado um subconjunto do metametamodelo MOF do OMG implementado na plataforma EMF (*Eclipse Modeling Framework*) (Eclipse Foundation, 2012), denominado Ecore, como metametamodelo que servirá de base para as técnicas de DDM

desenvolvidas neste trabalho.

2.1.2 TRANSFORMAÇÃO DE MODELOS

Os processos de transformação de modelos são divididos em duas categorias (CZARNECKI; HELSEN, 2006): Modelo para Modelo (M2M - *Model to Model*); e modelo para código (M2C - *Model to Code*). Os processos de transformação M2M têm o objetivo de gerar modelos (em geral mais próximos do espaço da solução) a partir de outros modelos (em geral mais próximos do espaço do problema). Os processos de transformação M2C, por outro lado, geram, a partir de modelos, representações concretas do espaço da solução, como trechos de código, documentos ou arquivos, por exemplo.

Neste trabalho foi adotada a categoria de processos de transformação M2M, possibilitando assim a geração de modelos independentes de plataforma (PIMs - *Platform Independent Models*). PIMs são artefatos intermediários para a geração de código e podem ser utilizados como entrada para transformações M2C. Neste trabalho, soluções já existentes na literatura foram adotadas para os processos de transformação M2C, como descrito no Capítulo 3. As vantagens de se gerar PIMs ao invés de código diretamente estão: (i) na possibilidade de se conseguir melhor manutenibilidade, e (ii) na propagação de mudanças realizadas nesses modelos, juntamente com as principais informações de domínio, para níveis mais concretos de projeto.

Existem diferentes abordagens para transformações M2M, sendo a especificação da OMG uma das mais utilizadas. Nessa especificação, um ou mais modelos de destino são gerados a partir de um modelo de origem, com base em um conjunto de regras de transformação, baseadas nos metamodelos de origem e de destino.

O arcabouço de qualquer processo de transformação de modelos, que segue as especificações do OMG, é ilustrado na Figura 2.2. Os modelos de origem (`Source Model`) e de destino (`Target Model`) possuem metamodelos associados (`Source Metamodel` e `Target Metamodel`), e as próprias regras de transformação estão associadas a modelos de transformação (`Transformation Model`) que são instanciados de um metamodelo correspondente (`Transformation Metamodel`). Além disso, todos os metamodelos (de origem, de destino e de transformação) são instâncias de especificações do metametamodelo MOF.

QVT (*Query View Transformation*) (OMG, 2008) e ATL (*Atlas Transformation Language*) (JOUAULT et al., 2008) são linguagens que permitem a definição de regras de

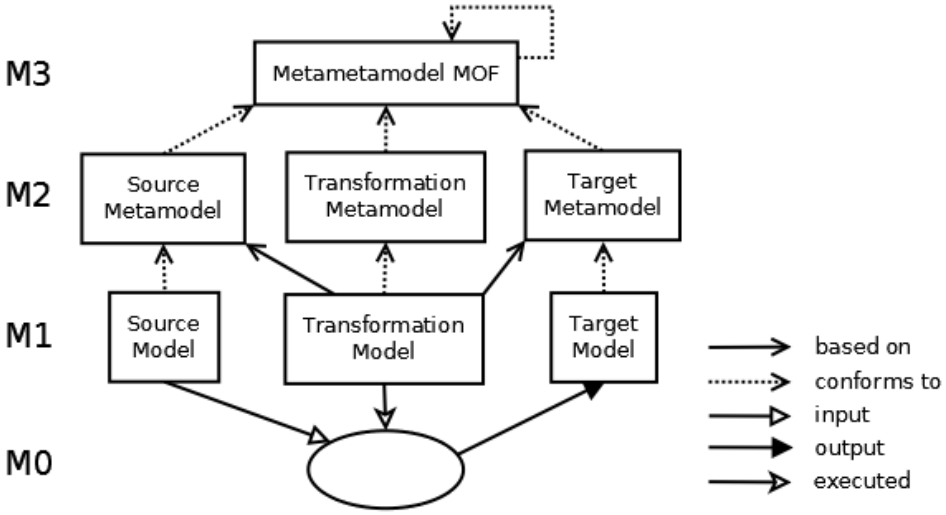


Figura 2.2: Transformação de modelos sobre o padrão da OMG (JOUAULT et al., 2008).

transformação e seguem as especificações da OMG. Neste trabalho foi adotada a linguagem ATL devido à sua maturidade de definições, documentação abrangente, e integração com o metametamodelo Ecore para a plataforma Eclipse. Devido ao fato de QVT ser estruturalmente semelhante a ATL, somente a linguagem ATL é apresentada de forma mais detalhada nesta seção.

ATL dá suporte à definição de regras declarativas (*matched rules*) e imperativas (*called rules*). *Matched rules* são desenvolvidas entre os metamodelos de origem e os metamodelos de destino. *Matched rules* possuem subtipos, como *standard rules* e *lazy rules*. *Standard rules* são aplicadas apenas uma vez a cada metadado encontrado no modelo de origem. *Lazy rules* são aplicadas a todos os elementos de um metadado a partir do modelo de origem. *Called rules* são regras imperativas que são invocadas por outras regras. A principal diferença entre *matched rules* e *called rules* é que esta última não possui metadados de origem e os metadados de destino são criados de forma imperativa. No presente trabalho, procurou-se empregar regras declarativas, pois as mesmas oferecem algumas vantagens: (i) destacam as relações entre metadados de origem e destino de forma intuitiva; (ii) omitem detalhes relacionados a seleção de metadados; e (iii) permitem o encadeamento e rastreio automáticos para outras regras.

A Figura 2.3 apresenta um exemplo simples de código ATL ("`ATLModelSample.atl`") na plataforma EMF, cuja sintaxe é explicada rapidamente abaixo:

- Na linha 1 é especificado o módulo da transformação de modelos através do cons-

trutor "*module*".

- Na linha 2 são especificados os modelos e metamodelos de origem e destino através dos construtores "*create*" e "*from*". O construtor *create* permite especificar variáveis referentes ao modelo ("`target`") e metamodelo ("`TargetMetamodel`") de destino. Por outro lado, o construtor *from* permite especificar variáveis referentes ao modelo ("`source`") e metamodelo ("`SourceMetamodel`") de origem. Essas variáveis são manipuladas nas regras de transformação.

- Nas linhas 4, 9 e 16, são especificadas regras de transformação do tipo *standard rules*, com metadados de origem e destino.

- Nas linhas 5, 10 e 17, são especificados os metadados de origem. Por exemplo, na linha 5, a variável "`source`" é uma variável para a instância do metadado "`Component`" do metamodelo de origem "`SourceMetamodel`".

- Nas linhas 6, 11 e 18, são especificados os metadados de destino. Por exemplo, na linha 6, a variável "`target`" é uma variável para a instância do metadado "`Composite`" do metamodelo "`TargetMetamodel`".

- Na linha 12 é especificado o relacionamento entre as regras "`Composite2Composite`" e "`Component2Component`". Todo relacionamento *children* do metadado de origem ("`source.children`") está vinculado ao relacionamento *children* ("`children`") do metadado de destino .

Regras de transformação de modelos na linguagem ATL podem ser estruturadas considerando padrões de projetos (GAMMA et al., 1995), viabilizando um melhor reuso e manutenibilidade das regras. Como ilustrado nas linhas 9 e 16 na Figura 2.3, as regras "`Composite2Composite`" e "`Leaf2Leaf`" foram estendidas ("`extends`") da regra "`Component2Component`", e esta última foi relacionada ("`matched`") com a primeira, configurando o uso do padrão *Composite*. A Figura 2.3 ilustra a presença desse padrão nessas regras por meio de sua representação correspondente em um diagrama de classes UML (pacote "`ATL Model`"). O diagrama de classes ilustra ainda como as estruturas de metadados dos metamodelos de origem (pacote UML "`Source Metamodel`") e destino (pacote UML "`Target Metamodel`"), que neste exemplo seguem o mesmo padrão *Composite*, são exploradas pelas regras. O resultado deste exemplo permite gerar a partir de um modelo

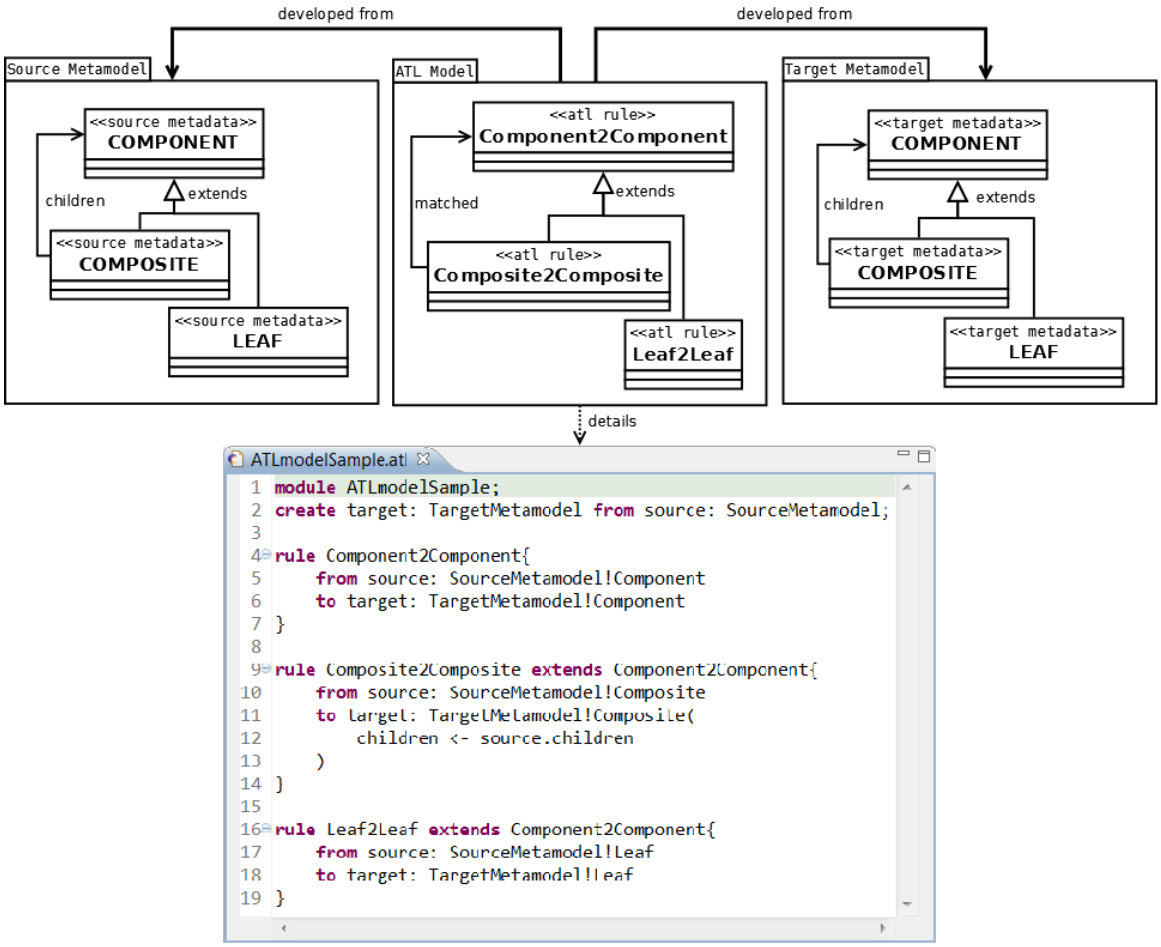


Figura 2.3: Exemplo do padrão de projeto *Composite* (GAMMA et al., 1995) aplicado em regras de transformação ATL (JOUAULT et al., 2008).

hierárquico de origem um modelo hierárquico de destino correspondente, com um número bastante reduzido de regras do tipo *standard rules*. É importante mencionar que regras declarativas do tipo *lazy rules* podem também ser estruturadas com padrões de projetos.

2.2 MODELAGEM DA INFORMAÇÃO NA SAÚDE

A eficácia no cuidado da saúde está relacionada com a capacidade e a qualidade da coleta, processamento e disseminação de informações médicas, o que é complexo de se obter considerando o alto dinamismo observado na área de saúde. Para citar um exemplo, observa-se uma mudança significativa no perfil epidemiológico da população em função da velocidade e complexidade da globalização, como pode ser visto durante a ocorrência de epidemias e pandemias, bem como na superlotação dos serviços de emergência (HOOT; ARONSKY, 2008). Mesmo com o auxílio de tecnologias como as TICs, o tempo necessário para res-

ponder a tais mudanças pode se estender por semanas ou até meses. A interoperabilidade de software na área da saúde tem sido proposta como uma estratégia para superar a situação atual, mas ainda existem obstáculos, considerando a complexidade da informação de domínio, envolvendo fatores locais, temporais e ontológicos (HUDSON; COHEN, 2010).

Fatores locais podem influenciar na complexidade do projeto de SIS. Dada uma área geográfica maior (por exemplo, uma área metropolitana), composta por um conjunto de áreas menores (por exemplo, municípios), pode-se observar que as áreas menores são geralmente desiguais em termos de tamanho da população, perfil demográfico, epidemiológico, condições de saúde, distribuição de renda, acesso a bens sociais (saneamento, abastecimento de água, emprego, etc.) bem como o acesso aos cuidados de saúde específico. Assim, a proximidade ou longitude de pequenas áreas não impede a existência de uma variação significativa da população que necessite de cuidados específicos de saúde. Dependendo do projeto do sistema de saúde, a resposta política a essas necessidades de saúde pode variar de uma pequena área para outra, gerando uma complexidade adicional.

Fatores temporais também podem influenciar na complexidade do projeto de SIS, o que está também associado à evolução tecnológica. A intensidade deste processo foi seguida pelo desenvolvimento de produtos na saúde, como equipamentos de diagnósticos, tecnologias terapêuticas e, em particular, padronização da informação na saúde. Em consequência, tem sido um desafio para os profissionais de saúde manter sua prática atualizada com a inovação de seu próprio campo de especialização. Uma das consequências do desenvolvimento tecnológico em ciências na saúde é a criação contínua de novos conceitos (e mudança dos conceitos previamente existentes), com reflexos na prática (CHRISTOPH, 2004).

A complexidade do fator ontológico é parcialmente derivada da dinâmica descrita anteriormente. No entanto, é uma característica específica da maneira que o conhecimento de saúde é criado e disseminado, o que aumenta a complexidade da definição de conceitos na saúde. Uma vez que a ciência na saúde está localizada frente às ciências biológicas, sociais e exatas, pode haver diferentes definições para um único conceito na saúde, ou seja, a probabilidade de obtenção de um consenso nos cuidados na saúde é bastante diminuto. A variabilidade de conceitos entre escolas de saúde, em conjunto com a variabilidade independente das políticas e culturas na saúde, entre outros fatores, pode diminuir o nível de consenso sobre um determinado conjunto de conceitos na saúde (SHALOM et al.,

2009).

Além disso, especialistas de um mesmo domínio tem necessidades de informações diferentes. Portanto, as aplicações ou, pelo menos, o ponto de vista sobre as aplicações, precisa ser muito específico a fim de melhorar a usabilidade. Grandes sistemas padronizados são geralmente lentos para se adaptar ao ritmo acelerado das mudanças ditadas pela adoção de novas tecnologias da saúde emergentes. Assim, os SIS não vão permanecer por muito tempo a menos que os padrões de interoperabilidade e terminologias possam evoluir rapidamente, o que é pouco factível dada às suas complexidades.

O resultado final da presença das complexidades de domínio em SISs, como as discutidas anteriormente, consiste em sistemas não interoperáveis e com alto custo de manutenção. Estas questões têm um impacto significativo sobre o baixo nível de adoção de TICs nos sistemas de saúde dinâmicos e emergentes em todo o mundo, em particular quando comparado a outros setores da economia global (MCGINN et al., 2011). Um dos caminhos para se tentar resolver em parte o problema é o uso de padrões emergentes.

2.2.1 PADRONIZAÇÃO DA INFORMAÇÃO NA SAÚDE

Registros Eletrônicos de Saúde (RESs) tem sido um campo de investigação fundamental em informática médica por muitos anos. Segundo a síntese de (EICHELBERG et al., 2005), a concepção de RESs tem o objetivo de modelar a saúde de um indivíduo digitalmente e apoiar a continuidade de seus cuidados, além de garantir a confidencialidade de dados. Interoperabilidade semântica é essencial para permitir o processamento de dados, o que é um pré-requisito para o apoio à decisão inteligente e planejamento da assistência, que são indiscutivelmente os mais importantes na adoção de RESs. A interoperabilidade de RESs pode contribuir de forma mais eficaz no suporte ao paciente, facilitando a recuperação e processamento de informações clínicas em diferentes locais. RESs interoperáveis permitem o transporte de informações entre diferentes SISs, o que pode acelerar os cuidados na saúde, além de evitar eventuais duplicatas. RESs podem reduzir erros, aumentar a produtividade e beneficiar o atendimento ao paciente.

Constantes mudanças nas informações em saúde afetam diretamente a interoperabilidade semântica e, consequentemente, o custo de desenvolvimento de SISs. Especificações padronizadas para informações na saúde como *open*EHR, HL7 e MLHIM definem modelos de dados, estruturas de dados e procedimentos para a modelagem do conhecimento e

representação de conceitos clínicos, o que é um requisito básico para se obter interoperabilidade semântica. Modelos de dados clínicos multinível como o *open*EHR e MLHIM, pensados para a concepção de RESs interoperáveis, podem ainda facilitar sobremaneira a aplicação de técnicas de DDM para geração de código de SISs, haja vista que esses modelos estratificam conceitos de tecnologia (alvos das transformações de modelos) e de domínio (conceitos clínicos definidos por especialistas). Neste trabalho foram adotadas as especificações *open*EHR devido à sua maturidade de definições e documentação abrangente. Devido ao fato das especificações MLHIM serem estruturalmente semelhantes ao *open*EHR, somente as especificações *open*EHR são apresentadas de forma mais detalhada nesta seção.

A fundação *open*EHR adota uma abordagem colaborativa, aberta e disseminada para a concepção de RESs (BEALE, 2002). As especificações *open*EHR têm o princípio que modelos de dados clínicos podem ser criados em dois níveis de modelagem, considerando a separação da informação e do conhecimento. O primeiro nível de modelagem é baseado em um "Modelo de Referência" (ou RM - *Reference Model*). O RM é em realidade um metamodelo que permite a modelagem da informação usando conceitos tecnológicos para maior interoperabilidade computacional. O segundo nível de modelagem é baseado no "Modelo de Arquétipos" (ou AM - *Archetype Model*). O AM é um outro metamodelo que define a modelagem de dados clínicos a partir de uma estrutura genérica (denominada arquétipo) para diferentes tipos de conhecimento clínico.

A abordagem de dois níveis de modelagem *open*EHR, descrita a seguir, é apresentada de forma simplificada na Figura 2.4. Essa ilustração toma como exemplo o arquétipo de "pressão arterial" do repositório *Clinical Knowledge Manager* (CKM, 2012), no ambiente de desenvolvimento Eclipse, além de diagramas de classes UML referentes aos metamodelos de informação (RM) e conhecimento (AM). Para descrever o exemplo ilustrado na Figura 2.4, é necessário descrever cada metadado envolvido.

- Metadados AM: Metadados com o prefixo "`C`" são nomeados desta forma, nas especificações *open*EHR, porque o metamodelo AM tem o objetivo de instanciar elementos com o conhecimento dos especialistas de domínio (esse conceito é dado como *restrições - Constraints*). Outro prefixo utilizado é o "`DV`" (originado de *Data Value*), que representa um metadado genérico para valores de dados. Importante notar que esse prefixo também é utilizado no metamodelo RM. Os metadados serão descritos

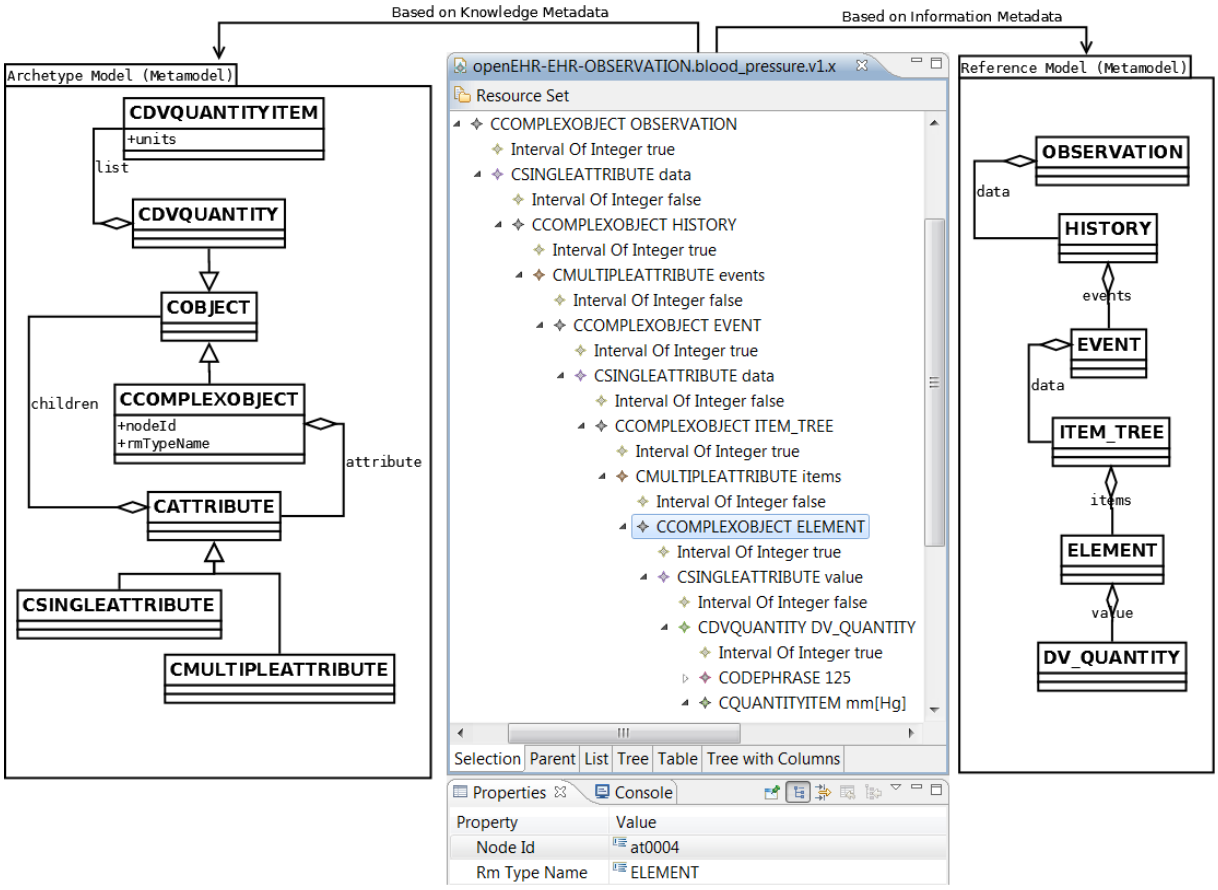


Figura 2.4: Exemplo de modelagem do arquétipo "pressão arterial" (CKM, 2012), considerando metamodelos de informação e conhecimento (BEALE, 2002).

a seguir, considerando a Figura 2.4: `COBJECT` é um dos metadados mais abstratos e genéricos que define objetos do domínio clínico de interesse; `CCOMPLEXTOBJECT` e `CDVQUANTITY` são metadados concretos de `COBJECT`, o metadado `CCOMPLEXTOBJECT` define e restringe objetos além de comportar uma lista de atributos derivados do metadado `CATTRIBUTE`; `CATTRIBUTE` define e restringe atributos de elementos e pode comportar elementos derivados de `COBJECT`; `CSINGLEATTRIBUTE` e `CMULTIPLEATRIBUTE` definem, respectivamente, um único atributo e múltiplos atributos. Por fim, `CDVQUANTITY` define e restringe elementos quantitativos (por exemplo, "sistólico" e "diastólico" são elementos para valores quantitativos para o conceito clínico "pressão arterial"), além de comportar elementos derivados de `CQUANTITYITEM` que pode especificar unidades de medidas ("`units`") (por exemplo, o elemento "sistólico" do conceito clínico "pressão arterial" pode ser especificado com unidades para medida em milímetros de mercúrio - "`mm[Hg]`").

- Metadados RM: Os metadados do RM podem especificar diferentes tipos de dados clínicos de forma abstrata, como tipos de modelos clínicos para "observação", "instrução", "avaliação" e "ação". Por simplicidade serão descritos somente os metadados para o tipo de dado clínico "observação", considerando a Figura 2.4: O metadado "`OBSERVATION`" define dados clínicos para o presente e o passado. Essa informação temporal pode ser registrada no relacionamento "`data`" para o metadado "`HISTORY`". O metadado "`HISTORY`" permite especificar distintos eventos temporais, essa informação pode ser registrada no atributo "`data`" ao metadado "`ITEM_TREE`". O metadado "`ITEM_TREE`" permite especificar elementos estruturados em árvore para informação na saúde a partir de relacionamentos ("`items`") com metadados, como "`ELEMENT`" que define elementos clínicos. Como ilustrado na Figura 2.4, o elemento "sistólico" foi atribuído com uma instância do metadado "`ELEMENT`" com a identificação "`at0004`" e o mesmo possui uma referência ("`value`") para uma estrutura quantitativa ("`DVQUANTITY`").

Como ilustrado na Figura 2.4, o dado clínico é modelado em dois níveis (informação e conhecimento). O especialista de domínio constrói arquétipos utilizando ferramentas de modelagem de arquétipos. Por exemplo, se o especialista deseja construir um arquétipo de "observação", o metamodelo RM será um "guia" para representar o conhecimento do especialista, que é expresso por metadados genéricos do metamodelo AM. A sintaxe desta modelagem são arquétipos com instâncias de metadados de informação (RM) com atributos referentes aos metadados do AM. Outro exemplo, ilustrado na Figura 2.4, é o da definição do elemento "`at0004`" (sistólico) como sendo do tipo "`CCOMPLEXOBJCT`" (do metamodelo AM) com uma referência a semântica desta informação, considerando o atributo "`rmTypeName`" com o valor "`ELEMENT`" (do metamodelo RM). Essa técnica de modelagem em dois níveis também é aplicada aos elementos restantes do arquétipo ilustrado na Figura 2.4.

2.3 MODELAGEM DE SISTEMAS DE INFORMAÇÃO NA SAÚDE

Como dito, abordagens para RESs, como as especificações HL7, *open*EHR e MLHIM, oferecem um futuro promissor para SIS, como melhor comunicação entre sistemas e legibilidade em prontuários clínicos. A adoção de RESs para SISs pode mitigar desafios para

o projeto, implementação e manutenção destes sistemas, o que pode resultar em sistemas mais confiáveis, além de fornecer conhecimento e orientação para proporcionar melhores cuidados na saúde.

Entretanto, como discutido na Seção 2.2, existem complexidades no projeto de SISs, considerando informação de domínio, envolvendo, por exemplo, fatores locais, temporais, ontológicos e a dinâmica entre estes fatores. Contudo, técnicas de arquitetura de software bem consolidadas podem capturar padrões arquiteturais similares, como controle e comunicação, entre diferentes sistemas. A definição de famílias (ou arquiteturas de referência) que agrupam formalmente tais sistemas com estruturas arquiteturais semelhantes pode simplificar o projeto desses sistemas por meio de reuso não só de código como de outros artefatos de desenvolvimento, como regras de transformação de modelos. Exemplos de famílias no contexto de SISs incluem:

- Emergência Pré-Hospitalar: sistemas dessa família têm o objetivo de levar, através de TICs, o conhecimento de um especialista até o local onde vítimas são socorridas imediatamente por emergencistas (ou *paramedic*, termo utilizado na literatura médica norte-americana) em unidades móveis (ambulâncias, helicópteros) ou de pronto atendimento geograficamente isoladas. Somente os emergencistas podem socorrer vítimas com graves lesões nos primeiros minutos do ocorrido, conhecido como "*golden hour*", o que aumenta as chances da vítima sobreviver, além de evitar sequelas irreversíveis. Com suporte de sistemas, os emergencistas podem esclarecer dúvidas que podem colocar a vítima em risco de morte. O domínio da Família de Sistemas para Emergência Pré-Hospitalar pode ser genérico o suficiente para subdomínios, como acidentes de trânsito e no trabalho, assim como vítimas de violência, queimadura, tentativa de suicídio e infarto agudo do miocárdio (IAM). Por exemplo, para vítimas de IAM, somente um cardiologista pode instruir um possível tratamento ou remoção da vítima para uma unidade coronariana.

- Vigilância Epidemiológica: sistemas dessa família têm o objetivo de coletar e monitorar dados clínicos, periodicamente, relacionados a doenças e outros problemas na saúde, considerando inconsistências, contradições e anomalias. Sistemas de vigilância epidemiológica possuem um potencial significativo para acelerar e antecipar a detecção de surtos epidêmicos (CHEN et al., 2010). A coleta de dados pode ser realizada por profissionais da saúde em departamentos de saúde, como salas

de emergência, ambulatórios, hospitais, entre outros. A monitoração de dados é analisada por epidemiologistas, através de dados clínicos individuais, para produzir diferentes avaliações consistentes para surtos epidêmicos, como dores crônicas, câncer, diabetes, entre outros. A análise de dados coletados, periodicamente, permite detectar, alertar e notificar eventos que necessitam de atenção iminente para surtos epidemiológicos. Esses sistemas são capazes de detectar anomalias na saúde pública e reagir rapidamente a uma epidemia.

Segundo (BRAGA et al., 2012) e (GOMES et al., 2012), uma forma de construir elementos de famílias de SIS, pelo menos nos primeiros estágios de desenvolvimento, pode ser realizada através do tipo do profissional que opera esses sistemas e o tipo da informação de saúde que esses profissionais utilizam, como tipos de dados clínicos "observação", "instrução" e "avaliação", da especificação *open*EHR:

- Na Família de Sistemas para Emergência Pré-Hospitalar identificam-se elementos arquiteturais, como "emergencista" e "especialista" que são tipicamente associados com tipos de dados clínicos de observação e instrução. Identifica-se também um elemento para "central de teleconsulta", que consiste no armazenamento e gerenciamento dos dados trocados entre emergencistas e especialistas, além de uma maior tendência a bidirecionalidade dos fluxos de dados envolvidos em tempo real.

- Na Família de Sistemas para Vigilância Epidemiológica identificam-se elementos arquiteturais, como "coletor" e "epidemiologista", que são tipicamente associados com tipos de dados clínicos de observação e avaliação. Identifica-se também um elemento para a "central de dados", que consiste no armazenamento de dados manipulados pelos coletores e epidemiologistas. Diferente dos sistemas de emergência pré-hospitalar para suporte remoto em tempo real, os sistemas de vigilância epidemiológica, tipicamente realizam a coleta de informações da população em períodos indeterminados de tempo, de forma unidirecional, até que seja identificada por um epidemiologista uma possível epidemia.

A construção de sistemas através de famílias de SIS permite não só reutilizar ativos de desenvolvimento como também satisfazer restrições arquiteturais. Por exemplo, não faz sentido utilizar dados clínicos de instrução no âmbito de oferecer suporte em tempo

real a sistemas de vigilância epidemiológica, pois os epidemiologistas tipicamente não têm interação com os pacientes.

2.4 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Este capítulo apresentou uma revisão da literatura considerando os principais conceitos e técnicas envolvidos na abordagem do trabalho proposto. Foram discutidos os conceitos relacionados a técnicas de Desenvolvimento Dirigido por Modelos (DDM), como metamodelos e transformação de modelos, além de concepções para modelagem da informação na saúde, como os padrões especificados pela fundação *open*EHR, bem como alguns tipos de SISs existentes.

3 SISTEMA DE TRANSFORMAÇÃO

Este capítulo apresenta o sistema de transformação IMT (*Interface Model Transformations*) cujo principal objetivo é a geração de código de GUI para SISs por meio do emprego de técnicas de Desenvolvimento Dirigido por Modelos (DDM). O sistema de transformação IMT recebe como entrada modelos de dados clínicos e o modelo arquitetural do SIS pretendido e gera como saída código de GUI.

O sistema de transformação IMT adota um subconjunto do metametamodelo MOF do OMG implementado na plataforma Eclipse, denominado Ecore (Eclipse Foundation, 2012). O metametamodelo Ecore servirá de base para a metamodelagem de modelos de dados clínicos e de modelos arquiteturais de SISs.

Os modelos e metamodelos de dados clínicos adotados no sistema de transformação IMT são especificações da organização *open*EHR (BEALE, 2002). O metamodelo correspondente ao modelo de referência (RM) do *open*EHR foi convertido para o formato Ecore, como detalhado na Seção 3.2.1. Os modelos de dados clínicos (arquétipos) utilizados nos exemplos deste capítulo (e também do Capítulo 4) foram coletados do repositório *Clinical Knowledge Manager* (CKM, 2012) da fundação *open*EHR, que é um sistema colaborativo para revisão, gerenciamento, discussão e validação de conceitos clínicos por especialistas de domínio. Os modelos e metamodelos para a definição de famílias de SISs também são definidos a partir do formato Ecore e são inspirados em algumas construções presentes em Linguagens de Descrição Arquitetural (ADL), como a linguagem Acme (GARLAN et al., 1997).

Como ilustrado na Figura 3.1, o sistema de transformação proposto possui três componentes principais: "`M2M Transformation`", "`M2C Transformation`" e "`Transformation Rules Repository`". Os dois primeiros são os componentes do sistema de transformação IMT que executam regras de transformação de modelos (CZARNECKI; HELSEN, 2006), respectivamente, para os processos de transformação Modelo para Modelo (M2M) e Modelo para Código (M2C) empregados neste trabalho. O terceiro elemento é um repositório de regras reutilizáveis de transformação de modelos. Esses elementos são detalhados a seguir.

O componente `M2M Tranformation` processa os modelos de dados clínicos ("`Clinical`

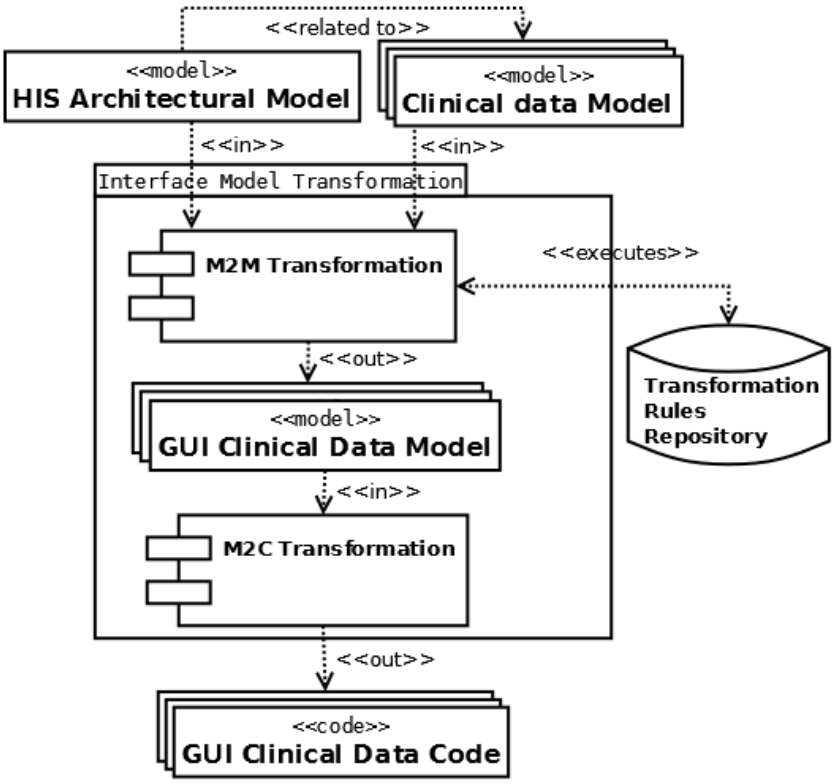


Figura 3.1: Sistema de transformação para Sistemas de Informação em Saúde.

`Data Model`") relacionados ao modelo arquitetural de SIS pretendido ("`HIS Architectural Model`"') e, a partir destes modelos, executa as regras de transformação localizadas no repositório de regras reutilizáveis ("`Transformation Rules Repository`"). A linguagem ATL (*Atlas Transformation Language*) (JOUAULT et al., 2008) foi adotada para o processo de transformações M2M. O resultado desse processo são modelos independentes de plataforma (PIMs) com as principais informações de domínio para GUI ("`GUI Clinical Data Model`"). O processo de transformações a partir do componente `M2C Transformation` realiza transformações dos modelos recebidos do componente `M2M Transformation` para modelos concretos de GUI ("`GUI Clinical Data Code`"). Essa transformação utiliza a ferramenta RichUbi (CIRILO et al., 2010) para gerar código em JSP (*JavaServer Pages*) e HTML (*HyperText Markup Language*), entre outros. Entretanto, é importante ressaltar que outras ferramentas para processos de transformações M2C podem ser utilizadas, bem como outras plataformas-alvo podem ser contempladas, uma vez que as principais informações de domínio, especializadas em GUI, estão contidas nos PIMs gerados pelo componente "`M2M Transformation`".

3.1 TRANSFORMAÇÕES DE MODELOS DE DADOS CLÍNICOS COM INFORMAÇÕES ARQUITETURAIS

Uma hipótese importante considerada neste trabalho é a de que o emprego de técnicas de DDM com vistas à geração de código de SISs sem considerar informações arquiteturais pode levar a uma explosão combinatória de regras de transformação, uma vez que cada novo SIS a ser gerado implica na especificação de um novo conjunto de regras. Como consequência, o custo de desenvolvimento é deslocado para a especificação das regras, o que demanda conhecimento especializado de linguagens cujas estruturas (em geral baseadas em conceitos de metamodelagem) não fazem parte da prática comum de desenvolvimento de software. Isso se reflete em última instância em um custo de desenvolvimento potencialmente mais alto que o não emprego de DDM. Para apoiar esta hipótese no contexto da geração de código de GUI, pode-se citar como exemplo as famílias de SISs de emergência pré-hospitalar e de vigilância epidemiológica, apresentadas no Capítulo 2, onde diferentes combinações de tipos de dados clínicos ocorrem devido a presença de distintas estruturas arquiteturais.

- Em uma família de SISs de suporte remoto a emergência pré-hospitalar, identificam-se dois componentes arquiteturais fundamentais relacionados a GUI: o "emergencista", que representa o primeiro atendimento ao paciente; e o "especialista", que auxilia remotamente o emergencista na tomada de decisão quanto ao procedimento clínico a ser adotado no atendimento. Um exemplo de SIS pertencente a essa família, focado no atendimento emergencial de vítimas de infarto, é descrito em (CORREA et al., 2011). Nessa família, dados de observação (como "pressão arterial") são tipicamente associados a "elementos de formulário" no componente emergencista e a "elementos de visualização" no componente especialista, enquanto o inverso acontece para dados de instrução (como "intervenção a ser adotada"). Identifica-se também um componente não relacionado a GUI, a "central de teleconsulta", que consiste no armazenamento e gerenciamento dos dados trocados entre os componentes "emergencista" e "especialista", portanto, este componente não possui propriedades de GUI.

- Em um sistema de vigilância epidemiológica, diferentes combinações ocorrem devido à presença de diferentes estruturas arquiteturais - um *survey* desses sistemas

é apresentado em (JOB A. T. A. GOMES, 2011). Contudo, de forma geral, há uma maior relevância de dados clínicos de observação e avaliação (como "clusters de casos de cólera"), em contraposição aos dados de observação e instrução dos sistemas de atendimento emergencial, além de uma maior tendência a unidirecionalidade dos fluxos de dados envolvidos. Na família de sistemas de vigilância epidemiológica são identificados dois componentes arquiteturais fundamentais relacionados a GUI: "coletor" e "epidemiologista". O primeiro componente realiza a coleta de dados clínicos de observação em elementos de formulários. O segundo componente permite a um epidemiologista analisar os dados coletados, portanto, este componente está relacionado a elementos de visualização. Identifica-se também um componente não relacionado a GUI, a "central de dados", que consiste no armazenamento de dados manipulados pelos componentes "coletor" e "epidemiologista", portanto este componente não possui propriedades de GUI. Diferente dos sistemas de emergência pré-hospitalar, onde os componentes trocam informações no suporte remoto em tempo real, os sistemas de vigilância epidemiologia tipicamente realizam a coleta de informações da população em períodos indeterminados de tempo até que seja identificada uma possível epidemia.

Desta forma, o sistema de transformação IMT emprega uma estratégia para geração de código de SISs em que o desenvolvedor é capaz de categorizar SISs em diferentes famílias e definir um conjunto de regras de transformação comum a todos os SISs de uma família. A representação das famílias de SISs foi inspirada em algumas construções presentes em ADLs como Acme. Essas construções são: propriedades, componentes e estilos arquiteturais. O sistema de transformação oferece à modeladores de SISs um estilo arquitetural base que descreve propriedades fundamentais associadas a GUIs, como "Elemento de visualização" e "Elemento de formulário", por exemplo. Essas propriedades permitem enriquecer componentes definidos em famílias de SISs específicas - estas também descritas como estilos arquiteturais derivados do estilo arquitetural base - com informações de GUI. Exemplos de componentes seriam: (i) "Emergencistas" e "Especialistas" no estilo arquitetural que descreve a família de sistemas de atendimento emergencial; e (ii) "Coletores" e "Epidemiologistas" no estilo arquitetural que descreve na família de sistemas de vigilância epidemiológica. O que caracteriza um SIS individual em cada uma dessas famílias é o conjunto de dados clínicos de interesse para esse sistema: por exemplo, a informação de

"Tônus muscular" é crucial em obstetrícia de emergência (para o método Apgar) mas não faz sentido no atendimento emergencial a infartados. Essa estratégia permite um melhor reuso das regras de transformação por parte do sistema de transformação IMT, uma vez que se pode desacoplar as regras responsáveis pela tradução das informações arquiteturais (reusáveis no contexto de uma família) daquelas responsáveis pela tradução do modelo de dados clínicos de interesse (reusáveis na medida em que atuam no nível de modelagem relacionado aos conceitos de tecnologia nas especificações *open*EHR).

É importante destacar que a adoção, neste trabalho, de técnicas de DDM implica na existência de duas classes distintas de desenvolvedores usuários do sistema de transformação IMT. O *desenvolvedor de famílias* define metamodelos que descrevem diferentes famílias de SIS bem como as regras de transformação que atuam sobre modelos de SIS dessas famílias, e publica os mesmos no repositório de regras reusáveis do sistema de transformação IMT (desenvolvimento *para* reuso ou engenharia de domínio (CLEMENTS; NORTHROP, 2002)). O *desenvolvedor de SISs* usa os metamodelos e regras de transformação correspondentes publicados no repositório pelo desenvolvedor de famílias, bem como metamodelos de dados clínicos de interesse publicados no CKM da fundação *open*EHR pelos especialistas de domínio, para descrever modelos de SISs. Esses modelos combinam informações arquiteturais com conceitos clínicos e são usados pelos desenvolvedores de SISs na geração de código de GUI para esses SISs (desenvolvimento *com* reuso ou engenharia de aplicações). Assim, o *desenvolvedor de famílias* deve ser capaz de especificar arquiteturas, empregando metamodelos genéricos para famílias de SISs. Por outro lado, o *desenvolvedor de sistemas* deve ser capaz de modelar SISs específicos com reuso das arquiteturas, com base em seu conhecimento de domínio.

As próximas subseções descrevem os metamodelos nos quais o sistema de transformação IMT se baseia e que são usados pelo desenvolvedor de famílias na criação de novos metamodelos de famílias e das regras de transformação correspondentes. No Capítulo 4 são apresentados exemplos de aplicação desses metamodelos na concepção de novas famílias de SISs bem como de código de GUI para sistemas dessas famílias.

3.2 METAMODELAGEM

A Figura 3.2 apresenta uma visão geral dos relacionamentos entre os níveis de meta-metamodelo, metamodelos e modelos do sistema de transformação IMT. Como pode ser

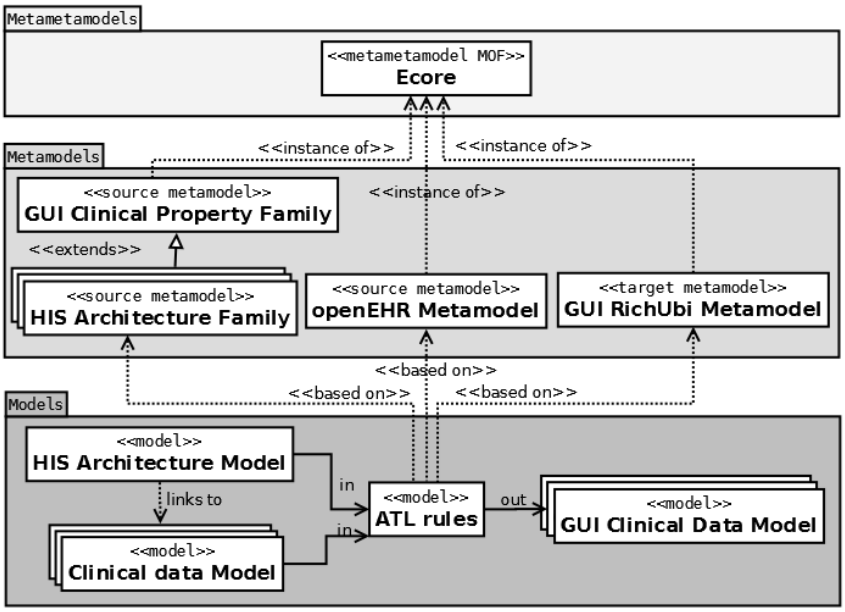


Figura 3.2: Relacionamento entre modelos e metamodelos para as transformações de modelos propostas.

visto no nível de modelos da figura, as regras ATL ("`ATL Rules`") têm como entrada os modelos de dados clínicos desejados ("`Clinical Data Model`") e o modelo arquitetural de SIS pretendido ("`HIS Architectural Model`"). Quando as regras são executadas, o componente `M2M Transformation` do sistema de transformação IMT gera modelos abstratos de GUI para dados clínicos ("`GUI Clinical Data Model`"). Todos os metamodelos necessários para o processo de transformações a partir do componente "`M2M Transformation`" foram definidos como instâncias do metametamodelo "`Ecore`". Esse relacionamento é necessário para o desenvolvimento e execução das regras de transformação pois a linguagem ATL requer a definição dos metamodelos sobre os quais ela opera no formato Ecore, garantindo interoperabilidade na plataforma *Eclipse Modeling Framework* (EMF) (Eclipse Foundation, 2012). O metamodelo "`GUI Clinical Property Family`" define o estilo arquitetural de base para propriedades de GUI. Esse metamodelo pode ser estendido por diferentes metamodelos (ilustrados na Figura 3.2 pelo conjunto "`HIS Architecture Family`"), cada um deles representando uma família de SISs específica. Por conta dessa derivação, os componentes arquiteturais definidos nessas famílias de SISs podem ser enriquecidos com propriedades de GUI. Esses metamodelos são discutidos com mais detalhes nas subseções 3.2.2 e 3.2.3. O metamodelo "`openEHR Metamodel`" define o metamodelo baseado em Ecore para as especificações *open*EHR, e é discutido com mais detalhes na

subseção 3.2.1.

3.2.1 METAMODELO ECORE PARA A ESPECIFICAÇÃO *OPEN*EHR

Considerando a estratégia proposta neste trabalho e considerando o sistema de transformação IMT, modelos de dados clínicos baseados na especificação *open*EHR, detalhada no Capítulo 2, devem ser transformados em modelos de GUI baseados no metamodelo RichUbi. Um problema na utilização do metamodelo *open*EHR é que o mesmo está disponível somente no formato de XMLSchema, o qual não é suportado pela linguagem ATL. Este problema pode ser resolvido por ferramentas de conversão de XML Schemas para Ecore (este trabalho utilizou ferramentas do Eclipse para esse propósito) (Eclipse Foundation, 2004). Através desta conversão, os arquétipos *open*EHR baseados em XML, disponibilizados pelo repositório de arquétipos *Clinical Knowledge Manager* (CKM, 2012) são interoperáveis tanto na especificação de XML Schemas, quanto na especificação Ecore.

Outra questão importante a ser ressaltada considerando a estratégia adotada é que cada componente arquitetural de famílias de SISs pode estar relacionado a um número indeterminado de modelos de dados clínicos, o que pode tornar o desenvolvimento de transformações na linguagem ATL complexo, uma vez que o desenvolvimento dessas transformações é restrito a um número fixo de modelos de origem. Esse problema foi solucionado a partir de uma modificação no metamodelo *open*EHR em Ecore para o mesmo ter uma funcionalidade extra para permitir múltiplos arquétipos em um único modelo.

Essa modificação foi realizada por meio do metadado EClass do metametamodelo Ecore no metamodelo *open*EHR (ilustrado na Figura 3.3 como "`ARCHETYPEGROUP`"). Através deste procedimento, um único modelo pode conter quantos modelos de dados clínicos (arquétipos) forem necessários para o SIS utilizado. A Figura 3.3 ilustra uma síntese de alguns arquétipos do repositório CKM, como "frequência cardíaca", "exame" e "avaliação de risco de condição", ilustrados, respectivamente nas identificações "`openEHR EHR OBSERVATION heart rate v1`", "`openEHR EHR OBSERVATION exam v1`" e "`openEHR EHR EVALUATION risk`", agrupados sob um mesmo "`ARCHETYPEGROUP`".

A modificação do metamodelo *open*EHR no formato Ecore, que permite múltiplos arquétipos em um único modelo, facilita o desenvolvimento das regras de transformação na linguagem ATL. Entretanto é necessário construir um modelo único de forma manual ou automática. Neste trabalho, a definição do modelo foi feita de forma manual. Essa

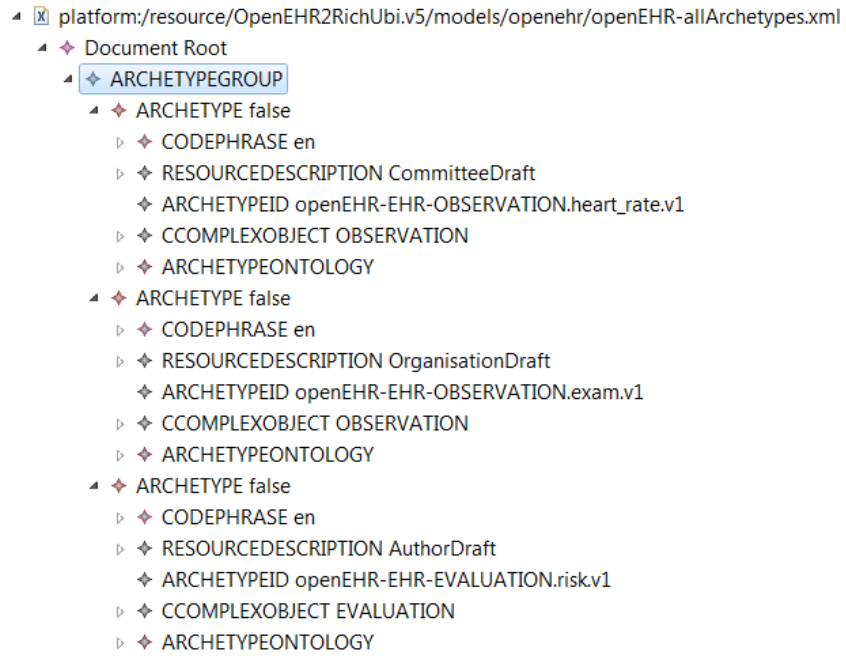


Figura 3.3: Modelo com alguns modelos de dados clínicos (arquétipos) para um determinado SIS.

construção deve ser orientada pelo especialista de domínio, para definir quais modelos de dados clínicos e seus respectivos elementos são de interesse para a geração de código GUI para um determinado SIS.

3.2.2 METAMODELOS ECORE PARA FAMÍLIAS DE SISS

No sistema de transformação IMT, considera-se somente os componentes arquiteturais de famílias de SISs para o desenvolvimento dos metamodelos referentes a estas famílias. A especificação arquitetural completa de famílias de SISs (incluindo mecanismos de interação entre componentes e restrições na composição entre os mesmos) está fora do escopo deste trabalho. No Capítulo 5 é discutida a aplicação do sistema de transformação IMT em uma linha de produto de software onde especificações arquiteturais completas são vislumbradas.

Neste trabalho, a construção de famílias de SISs é inspirada em alguns construtores presentes em ADLs, como os encontrados na linguagem Acme. Contudo, como mencionado anteriormente, a definição dos metamodelos para famílias de SISs deve estar no formato Ecore para permitir interoperabilidade através da linguagem ATL com a plataforma Eclipse. Por isso, este trabalho se baseou em uma simplificação do metamodelo no formato Ecore para a linguagem Acme apresentado por Baudry (2006). Esta simplifi-

cação incorpora o construtor para componentes arquiteturais originalmente proposto por Baudry, e define construtores simplificados (inspirados no trabalho de (BAUDRY, 2006)) para sistemas, propriedades e estilos arquiteturais. A razão dessa simplificação decorre do escopo do sistema de transformação IMT em gerar código de GUI somente, uma vez que, no metamodelo Ecore para a linguagem Acme apresentado por (BAUDRY, 2006), esses outros construtores tratam de questões arquiteturais diversas, como mecanismos de interação e restrições na composição entre componentes.

A Figura 3.4 exemplifica o uso de alguns desses construtores na especificação de uma família simples de sistemas que seguem o estilo arquitetural Cliente-Servidor (GARLAN et al., 1997). Um pacote Ecore é usado para conter os elementos arquiteturais de uma família ("`Domain Family Package`"), incluindo componentes arquiteturais e sistemas. Para definir, formalmente, componentes arquiteturais de famílias de SISs, deve ser utilizado o construtor "componente" ("`Component`") do metamodelo de Acme em Ecore proposto por Baudry ("`ADL`"). O construtor "sistema" ("`System`") agrupa os componentes arquiteturais da família ("`Client Component`" e "`Server Component`") e permite a especificação de SISs que seguem a estrutura definida por essa família.

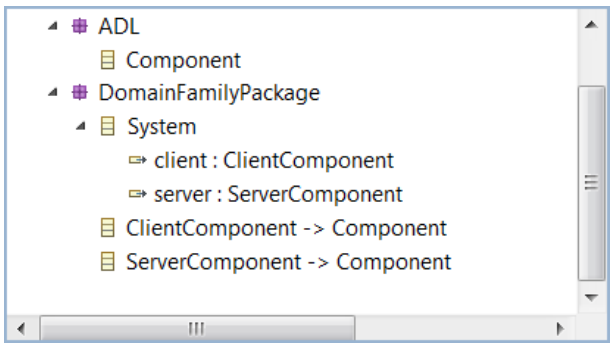


Figura 3.4: Família de Sistemas, no formato Ecore.

Considerando a abordagem para definição de famílias de SISs no formato Ecore, inspiradas em alguns construtores de ADLs, pode-se especificar componentes arquiteturais para diferentes famílias de SIS, como as de emergência pré-hospitalar e vigilância epidemiológica, discutidas por (BRAGA et al., 2012) e (GOMES et al., 2012) e usadas como exemplos no Capítulo 4.

3.2.3 METAMODELO ECORE PARA PROPRIEDADES DE GUI

Os componentes arquiteturais de famílias de SISs descritos na subseção anterior podem ser enriquecidos com metadados concretos que definem propriedades de GUI relacionadas com tipos de dados clínicos. Essa abordagem é ilustrada por meio de um diagrama de classes UML na Figura 3.5. Como ilustrado, o estilo arquitetural de base para propriedades de GUI ("`GUI Clinical Property Family`") possui uma classe abstrata ("`Property EHR`") com um atributo ("`archetypesIds`") que permite inserir uma lista de identificações para os modelos de dados clínicos (como o arquétipo de "Pressão arterial", que possui a identificação "`openEHR EHR OBSERVATION blood pressure v1`").

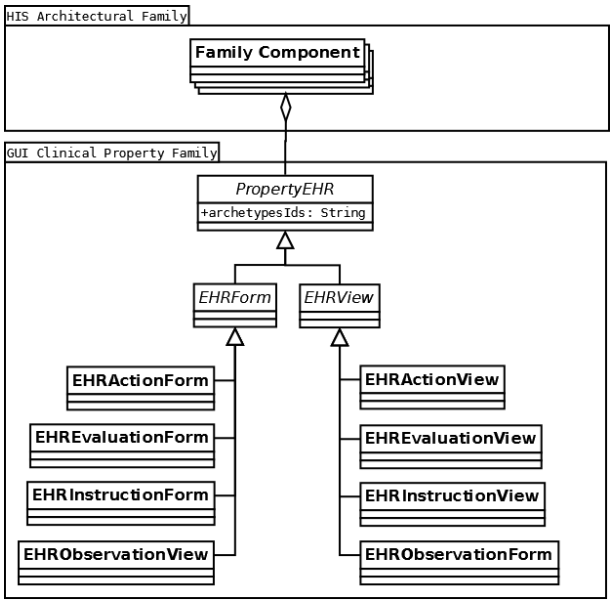


Figura 3.5: Família de propriedades de GUI considerando tipos de dados clínicos *open*EHR.

A Figura 3.5 ilustra ainda as demais classes de propriedades, que consistem em: (a) duas classes abstratas que definem formulário ("`EHR Form`") e visualização ("`EHR View`") - essas classes abstratas foram estendidas para herdar características mais abstratas da classe "`Property EHR`"; e (b) classes concretas que definem formulários e visualização de GUI considerando tipos de dados clínicos (como as classes "`EHR Observation Form`", "`EHR Observation View`", etc.). Novamente, essas classes concretas foram estendidas para herdar características das classes mais abstratas ("`EHR Form`", "`EHR View`" e "`Property EHR`"). Como ilustrado na Figura 3.5, componentes específicos de famílias de SISs (representados pelo conjunto "`Family Component`" - um exemplo de componente poderia ser o "`Client`

`Component`" da Figura 3.4) podem ser associados a classes concretas de propriedades de GUI para atribuir, de forma particular, informações extras de GUI considerando tipos de dados clínicos.

A Figura 3.6 ilustra o metamodelo de propriedades de GUI proposto desenvolvido no formato Ecore, que é ilustrado no pacote "`GUI Property`". A Figura 3.6 ilustra ainda um exemplo de uso do metamodelo de propriedades de GUI associado aos metamodelos de famílias de SISs, considerando o exemplo do estilo arquitetural Cliente-Servidor (pacote "`Domain Family Package`") apresentado anteriormente na Figura 3.4. Como se pode observar na Figura 3.6, os componentes de famílias de SISs (como "`Client Component`") podem ser relacionados com diferentes propriedades de GUI (como "`EHR Observation Form`") de acordo com o interesse do desenvolvedor de famílias em representar restrições na associação entre determinados tipos de dados clínicos, distintas formas de representação de GUI e diferentes componentes arquiteturais. Desse modo, o metamodelo de GUI tem pouco ou nenhum impacto nos metamodelos de famílias de SIS, o que é um ponto-chave na estratégia adotada para este trabalho de desacoplar os modelos arquiteturais dos modelos de dados clínicos, propiciando maior reuso das regras ATL.

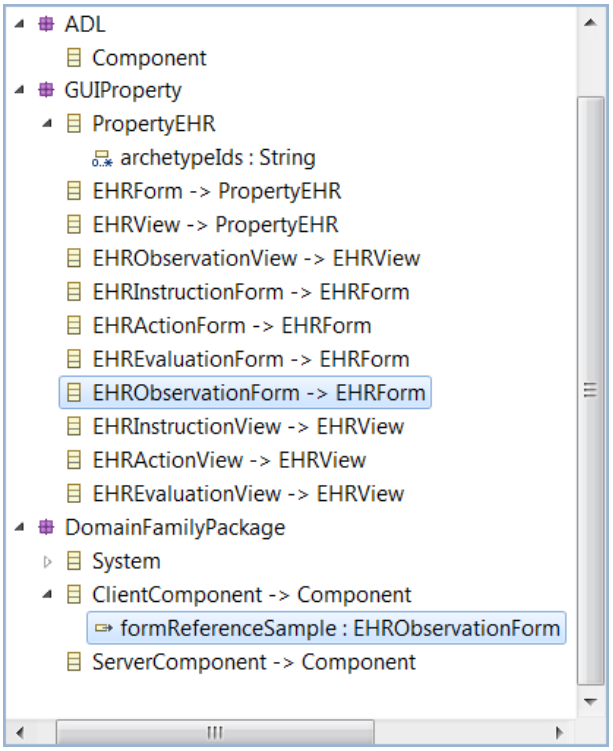


Figura 3.6: Definição de uma família de SISs considerando propriedades GUI e componentes ADL na especificação Ecore.

3.2.4 METAMODELO ECORE PARA GUI RICHUBI

Modelos de dados clínicos, considerando a especificação *open*EHR, possuem uma extensa hierarquia de informações, modeladas a partir das estruturas dos metadados que compõem o metamodelo *open*EHR. A "renderização" dessas informações em um SIS, seja como elementos de formulário ou de visualização, deve ser a mais próxima possível daquela pretendida pelos especialistas de domínio. Isso se deve fundamentalmente pela necessidade de se mitigar o problema da transcrição do conhecimento médico acerca dos dados clínicos relevantes em um SIS. Por isso, os processos de transformação do componente `M2M Transformation` do sistema de transformação IMT precisam reproduzir de forma mais fiel possível a hierarquia de informações dos modelos de dados clínicos em modelos de GUI.

O metamodelo RichUbi (CIRILO et al., 2010), especificado no formato Ecore, foi adotado neste trabalho para a definição dos modelos abstratos de GUI e sua consequente transformação M2C em código de GUI para SIS. O metamodelo e o processo de transformação de M2C das especificações RichUbi foram escolhidos porque permitem a construção de interfaces ricas, sensíveis ao contexto, e adaptáveis em tempo de execução quando visualizadas em diferentes dispositivos. O metamodelo RichUbi define o metadado `DIV` para o aninhamento de elementos de GUI, sendo portanto central na reprodução da estrutura hierárquica da especificação *open*EHR. Contudo, o metadado `DIV` não possui referência a "elementos de formulários" no metamodelo RichUbi, somente a "elementos de visualização". Por essa razão, foi desenvolvido neste trabalho uma modificação do metamodelo RichUbi para o mesmo permitir a reprodução completa da hierarquia de informações dos modelos de dados clínicos do *open*EHR.

3.3 REPOSITÓRIO PARA REGRAS DE TRANSFORMAÇÃO

Outro componente importante do sistema de transformação IMT é o Repositório de Regras de Transformação. A definição das regras de transformação é fundamental para o componente `M2M Transformation`, responsável pelos processos de transformação M2M. Essas regras foram estruturadas de acordo com a hierarquia dos metamodelos de origem e são executadas em sequência e de forma interativa, como detalhado adiante. Por questões de legibilidade, as regras são apresentadas sob a forma de diagramas de classe UML. A

descrição completa dessas regras na linguagem ATL é apresentada no Apêndice A deste trabalho.

A estrutura de regras no repositório é apresentada na Figura 3.7, através de modelos UML: os nomes das classes representam os nomes das regras; os atributos das classes representam as palavras reservadas "*from*" e "*to*" da linguagem ATL, os quais indicam, respectivamente, os metadados de origem e destino. Com o objetivo de organizar a definição das regras no repositório, foi aplicado o padrão de projeto *Composite* (GAMMA et al., 1995). Os estereótipos "<<`leaf rules`>>", "<<`composite rules`>>" e "<<`component rules`>>" ilustram a aplicação do padrão e os estereótipos "`extends`" e "`matched`"/"`calls`" ilustram o uso de herança e relacionamento, respectivamente. Por fim, funções auxiliares da linguagem ATL foram ilustradas no estereótipo "<<`helper`>>".

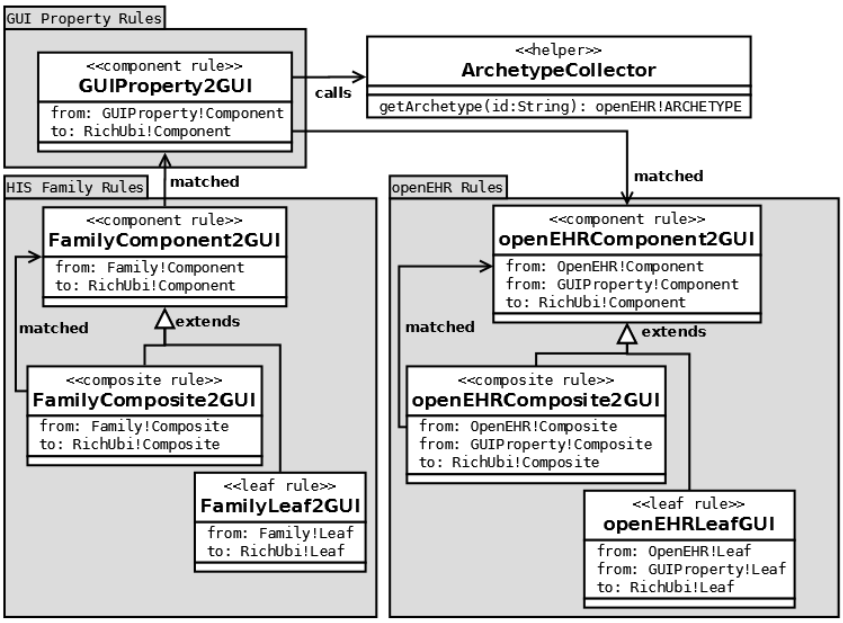


Figura 3.7: Grupos de regras de transformação propostos, considerando padrões de projetos.

As regras propostas foram estruturadas em três grupos principais descritos abaixo:

- O primeiro grupo de regras ("`HIS Family rules`") gera estruturas básicas para os modelos de GUI a partir de modelos arquiteturais. Essas estruturas básicas para modelos de GUI são metadados de GUI do RichUbi tais como "*portal*", "*document*", "*tab*", "*div*" entre outros. Este grupo de regras foi desenvolvido a partir de *standard rules* (vide subseção 2.1.2) que processam individualmente cada componente das famílias de SISs associado a propriedades de GUI. Foram consideradas nessas trans-

formações apenas as transformações de componentes de famílias SISs em portais *web* empregando o metadado `Portal` do metamodelo RichUbi, uma vez que está fora do escopo deste trabalho a definição completa de regras de transformação de estilos arquiteturais para famílias de SISs, como apontado anteriormente na seção 3.2.2.

- O segundo grupo de regras ("`GUI Property Rules`") foi definido a partir de *standard rules* que geram individualmente regiões de conteúdo para cada elemento de formulário e de visualização de acordo com as propriedades de GUI. Por exemplo, as propriedades de GUI para "visualizações para observação" e "visualizações para instrução", modeladas no componente de uma família de SISs, são transformadas em elementos de visualização, como o metadado "*div*" do metamodelo RichUbi, enquanto que "formulários para observação" e "formulários para instrução" são transformados para elementos de formulários, como o metadado "*form*" do metamodelo RichUbi. A regra mais abstrata deste conjunto utiliza funções auxiliares (*helper* ATL) "`Archetype Collector`" e, por parâmetros, envia o tipo de propriedade GUI ao grupo de regras "`openEHR rules`", responsáveis pelas transformações dos modelos de dados clínicos, descritas adiante. É importante destacar o papel central desse segundo grupo de regras na estratégia adotada para este trabalho. Esse segundo grupo de regras funciona como um *binding* entre o primeiro e o terceiro grupo de regras, permitindo que esses grupos evoluam de forma desacoplada. Dessa forma, novas regras para novas famílias de SIS (primeiro grupo) podem ser criadas sem a necessidade de criação de novas regras para transformação dos modelos de dados clínicos (terceiro grupo), implicando em um melhor reuso destas últimas.

- O terceiro grupo de regras ("`openEHR rules`") é responsável pelas transformações dos modelos de dados clínicos e seus respectivos elementos de acordo com as propriedades de GUI. Essas transformações consistem em reproduzir elementos de formulários (como os metadados "*list box*" e "*text area*" do metamodelo RichUbi) ou elementos de visualização (como o metadado "*span*" do metamodelo RichUbi). Essas regras também foram desenvolvidas considerando o padrão de projeto *Composite* com o objetivo de preservar as informações hierárquicas dos modelos de dados clínicos. As regras deste grupo foram especificadas como *lazy rules* para processar os modelos de dados clínicos quantas vezes forem encontrados. É importante frisar

que os dois primeiros grupos de regras possuem um metadado de origem *from*, enquanto este terceiro possui dois metadados de origem *from*. Esta técnica permite o desenvolvimento de distintas regras para cada combinação entre as propriedades de GUI ("visualizações para observação", "visualizações para instrução", "formulários para observação", "formulários para instrução", etc.) e elementos de dados clínicos, permitindo gerar distintos modelos de dados clínicos de GUI.

Os grupos de regras apresentados anteriormente foram discutidos de forma abstrata para melhor realçar o relacionamento entre as regras. As regras do terceiro grupo, por serem mais complexas, merecem uma atenção maior conforme apresentado abaixo.

A Figura 3.8 apresenta com maior detalhamento o terceiro grupo de regras de transformação de modelos de dados clínicos ("`openEHR rules`"), em classes UML, baseada na síntese das regras desenvolvidas na linguagem ATL, considerando o padrão de projeto *Composite*. Esse padrão é identificado no metamodelo *open*EHR e permite capturar "elementos estruturais" e "elementos de interesse dos especialistas de domínio". Os elementos estruturais podem ser relacionados a informações gerais ao modelo de dado clínico (metadado `ARCHETYPE`); ao tipo do elemento de interesse, como "observação" e "instrução" (metadados `CCOMPLEXOBJECT` e `COBJECT`); e a atributos com outras informações do elemento de interesse, como "protocolo" (metamodelo `CATTRIBUTE`). Os elementos de interesse dos especialistas de domínio, como "sistólico", "tamanho da braçadeira" e "comentário" do modelo do dados clínico "Pressão arterial" são definidos, respectivamente, nos metadados "`CDVQUANTITY`", "`CCOMPLEXTOBJECT ('DV CODED TEXT')`" e "`CCOMPLEXOBJECT ('DV TEXT')`".

A Figura 3.8 ilustra ainda regras, com o comentário UML, que tem como objetivos gerar elementos básicos de GUI a partir dos "elementos estruturais" ("`Generate basic elements`") e elementos de formulários e visualização a partir dos "elementos de interesse dos especialistas de domínio" (respectivamente, "`Generate view elements`" e "`Generate form elements`"). O ponto chave do grupo de regras para transformações de modelos *open*EHR é que as regras com comentários UML são executadas considerando o tipo da propriedade de GUI ("`EHRForm`" ou "`EHRView`"). Por exemplo: se o elemento "sistólico" do modelo de dados clínicos "Pressão arterial" estiver atribuído à propriedade de formulário, o mesmo será transformado em "elementos de formulário" ("`FieldSet`"); em contraposição, se estiver atribuído à propriedade de visualização, será transformado em "elementos de

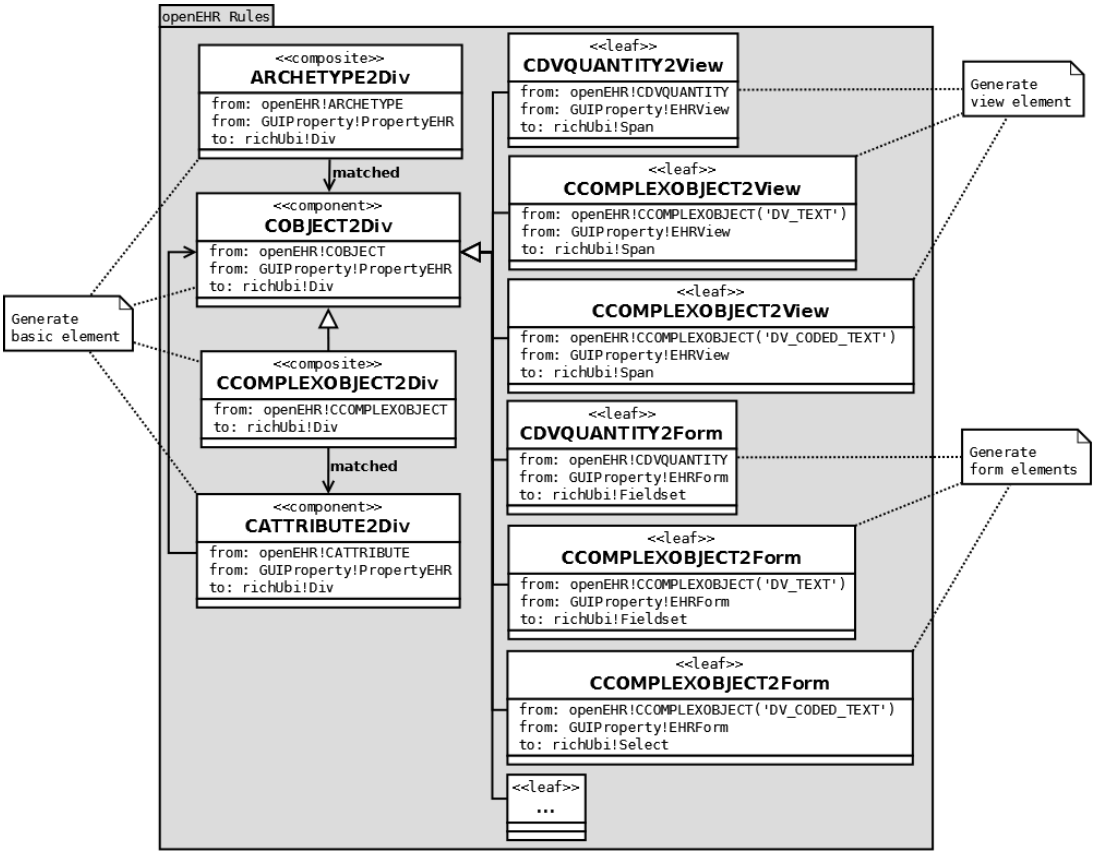


Figura 3.8: Síntese das regras de transformação de modelos de dados clínicos *open*EHR para GUI.

visualização" ("`Span`"). Desta forma, o mesmo ocorre para os outros elementos de dados clínicos.

As regras apresentadas acima são executadas considerando o polimorfismo da Orientação a Objetos (MARTIN; ODELL, 1998). Essas regras têm o objetivo de reproduzir a hierarquia de informações dos elementos dos modelos de dados clínicos de forma automática e interativa, permitindo maior reuso no desenvolvimento e na sua execução. No Capítulo 4 deste trabalho alguns exemplos detalhados da aplicação do sistema de transformação IMT serão apresentados.

3.4 TRABALHOS RELACIONADOS

Considerando o sistema de transformação IMT e a abordagem proposta para sua utilização, existem alguns trabalhos na literatura que adotam estratégias semelhantes.

A abordagem de (SCHULER et al., 2006) tem por objetivo gerar código de GUI considerando Registros Eletrônicos de Saúde (RESs) para garantir maior interoperabilidade

semântica para SISs. Essa abordagem considera modelos de dados clínicos, assim como o modelo de referência da fundação *open*EHR e também considera a especificação do HL7v3 (HL7 CDA, 2005). Segundo os autores, a especificação HL7v3 não foi utilizada porque não é possível separar de forma clara conceitos de tecnologia e de saúde em modelos de dados clínicos, diferente das especificações *open*EHR que permitem a construção de modelos de dados clínicos (informação de domínio) a partir de restrições de tipos de dados definidos em um modelo de referencia mais genérico (informação de tecnologia). Essa abordagem pode comprovar que o uso de uma linguagem baseada em XML para descrição de GUI pode trazer benefícios para a geração de código de GUI a partir de vários modelos de dados clínicos na especificação *open*EHR. Entretanto, essa abordagem não emprega transformações de modelos considerando um metametamodelo comum, como Ecore. A falta de técnicas de transformações de modelos formalmente especificadas e padronizações conceituadas como da organização OMG pode ser uma limitação para o desenvolvimento de SISs, considerando manutenibilidade, interoperabilidade e reuso. Transformações de modelos, como as viabilizadas pela linguagem ATL, podem resultar em diferentes modelos abstratos independentes de plataforma como o proposto pelo sistema de transformação IMT, diferente da abordagem de (SCHULER et al., 2006), que tem o objetivo de gerar somente código web específico para uma dada plataforma, o que compromete o reuso durante a fase de projeto do software.

A abordagem de (NARDON et al., 2007) explora a construção de SISs baseada nas especificações *open*EHR, como modelos de dados clínicos (arquétipos). Essa abordagem apresenta uma discussão sobre a complexidade de uso do *open*EHR no desenvolvimento de SISs, como a geração de código de GUI. Entretanto, não foi explorado nessa abordagem técnicas de metamodelagem considerando os benefícios de DDM para produção em larga escala, utilizando modelos independentes de plataforma e considerando transformações M2M e M2C. Essa limitação pode resultar em SISs com alta taxa de manutenção, pois a informação na saúde sofre constantes mudanças e pode afetar diretamente o código de GUI, o que leva a um maior esforço de desenvolvimento.

(ATALAG; YANG, 2010) exploram a geração de código GUI adaptativa a partir de modelos de dados clínicos (arquétipos) da especificação *open*EHR. Seu resultado é uma GUI adaptativa que permite ao especialista de domínio modificar a interface conforme sua preferência de visualização destas informações. O principal objetivo da abordagem é

gerar código de GUI mais amigável ao especialista de domínio. Nessa abordagem foram desenvolvidos modelos de dados clínicos (arquétipos) específicos para domínio na especificação *open*EHR. A criação destes modelos teve o apoio de especialista de domínio, que pôde validar a abordagem para a geração de código GUI adaptativo. Entretanto, como na abordagem de (NARDON et al., 2007), não foi explorado técnicas de DDM, como metamodelagem e transformação de M2M e M2C, para a produção em larga escala de sistemas. Isso pode resultar em baixa manutenibilidade e alto custo de desenvolvimento, considerando as constantes mudanças de informação na saúde.

A abordagem proposta por (LINDEN et al., 2009) explora a geração de código de GUI, em um alto nível de abstração, capaz de exibir dados clínicos indisponíveis de forma significativa para especialistas de domínio, considerando RESs, como HL7v3 e *open*EHR. Foi discutido também que a especificação *open*EHR sofre menos impacto na geração de código de GUI proposta, pois a especificação *open*EHR define um modelo de referência (metamodelo) em alto nível de abstração que não é preciso ser modificado, e os modelos de dados clínicos (arquétipos) são construídos por restrições considerando os tipos de dados definidos no modelo de referência. Essa abordagem permite que código de GUI possa ser gerado a partir de modelos de dados clínicos indisponíveis, considerando apenas os metamodelos de dados clínicos, como o metamodelo *open*EHR. Essa abordagem tem o mesmo principio do sistema de transformação IMT, que permite gerar códigos de GUI considerando o metamodelo *open*EHR para qualquer modelo de dado clínico (arquétipos). Por outro lado, o sistema de transformação IMT é limitado às especificações *open*EHR, enquanto que (LINDEN et al., 2009) pode tratar outras padronizações de RESs, como HL7v3. Entretanto, o sistema de transformação IMT considera técnicas de transformações de modelos formais, considerando padronizações da organização OMG. Essas técnicas foram exploradas no sistema de transformação IMT considerando outras informações que podem influenciar diretamente a geração de código, como diferentes estilos arquiteturais, que permitem a descrição de uma família de propriedades que pode ser especializada para contemplar diferentes famílias de SISs. Essas informações arquiteturais podem oferecer benefícios para o desenvolvimento de SISs em larga escala.

(COSTA, 2011) propõe uma abordagem para a construção e geração de código automática de GUI para sistemas de informação. Essa abordagem permite ao especialista de domínio (não restrito a saúde) interagir durante a construção de GUI. Esta aborda-

gem considera metamodelagem e regras de transformação de modelos em um alto nível de abstração. O trabalho explora também o desacoplamento de informações distintas de GUI, definindo novos metamodelos para distintos domínios para construção de GUI em sistemas de informação. Isso resulta em maior flexibilidade e em mudanças no processo geral de projeto de software. . Entretanto, essa abordagem não se baseia em um metametamodelo comum (como MOF ou Ecore) tanto para os metamodelos como para as regras, o que compromete tanto o reuso de projeto como a interoperabilidade do ferramental resultante. Além disso, a ausência de tratamento específico para modelos de dados clínicos torna inviável a geração de código sem perder informações de domínio.

A abordagem de (MENÁRGUEZ-TORTOSA et al., 2011) explora geração de código completo para SISs considerando especificações de RESs, como *open*EHR e HL7v3. Essa abordagem tem o objetivo de atender a falta de ferramentas para o uso e aplicação de tais especificações. Essa abordagem considera técnicas de DDM, como processos de transformações M2M e M2C, como proposto neste trabalho aplicado ao sistema de transformação IMT. (MENÁRGUEZ-TORTOSA et al., 2011) consideram geração de código através de formulários web para preenchimento de dados clínicos que são enviados por mensagens padronizadas, tornando possível a alimentação de bases de dados que seguem esse padrão, garantindo interoperabilidade por construção. Entretanto, essa abordagem não explora a questão de que diferentes famílias de SISs demandam diferentes tratamentos para os dados clínicos em diferentes componentes de um SIS, o que só é possível de ser feito quando os dados clínicos são enriquecidos com informações arquiteturais, como proposto neste trabalho.

Em resumo, muitos dos trabalhos apresentados acima estão focados no desenvolvimento de SISs para as necessidades da população de forma imediata que podem usufruir da tecnologia e métodos disponíveis. Técnicas de DDM podem trazer benefícios para esses sistemas, uma vez que permitem a produção de software em larga escala e com alto nível de manutenibilidade, interoperabilidade e reuso. Outros trabalhos encontrados exploram esses benefícios com maior nível de abstração de modelagem, entretanto, são abordagens complexas de serem implementadas no cenário atual da saúde, tanto em grandes países ou pequenas regiões.

3.5 CONSIDERAÇÕES FINAIS DO CAPÍTULO

O sistema de transformação IMT oferece vantagens ao adotar processos de transformação M2M (`M2M Transformation`), pois permite a geração de modelos independentes de plataforma. O resultado deste processo possibilita o uso de outros processos de transformação M2C (`M2C Transformation`), além dos que a ferramenta RichUbi oferece. Outra vantagem da abordagem proposta é a forma como se procedeu com a metamodelagem aplicada nos processos de transformação M2M, permitindo a definição de grupos de regras de transformação desacoplados para modelos arquiteturais e modelos de dados clínicos. Isso permite maior reuso de regras, além da execução automática destas regras, o que pode tratar distintas possibilidades de transformações de modelos de dados clínicos para modelos de GUI, considerando informações arquiteturais.

Uma limitação da abordagem é que para cada nova família de SISs é necessário criar novas regras de transformações. Espera-se que o esforço de desenvolvimento das regras não precise ser reproduzido para cada novo sistema constituinte de uma família, mas isso implica na necessidade de se explorar até que ponto a definição de uma família (e as regras correspondentes) pode ser de fato reusada. Além disso, foi considerado que o esforço de desenvolvimento pode ser reduzido se as novas regras forem aplicadas na definição da regras de transformação utilizando a família de propriedades de GUI. O desenvolvimento de novas regras pode sofrer pouca ou nenhuma modificação considerando o metamodelo de propriedades.

4 APLICAÇÃO DO SISTEMA DE TRANSFORMAÇÃO

Este capítulo apresenta cenários de utilização do sistema de transformação IMT, para a geração automática de código GUI para SISs, a partir da síntese de modelos de dados clínicos (arquétipos) combinados a modelos arquiteturais. Serão apresentados exemplos de uso para a especificação de metamodelos Ecore, considerando duas famílias de SISs (BRAGA et al., 2012; GOMES et al., 2012). Para o primeiro cenário de uso foi especificada uma família de sistemas de emergência pré-hospitalar e para o segundo cenário de uso uma família de sistemas de vigilância na saúde. É importante ressaltar que nestes cenários considerou-se apenas os componentes arquiteturais destas famílias.

4.1 DEFINIÇÃO DOS METAMODELOS ARQUITETURAIS

A primeira etapa na especificação dos cenários de uso utilizando o sistema de transformação IMT envolve o desenvolvedor de famílias, que tem o papel de definir metamodelos arquiteturais com as informações necessárias para o processamento das transformações de modelos de dados clínicos. A definição desses metamodelos engloba: (a) metamodelos no formato Ecore para famílias de SISs; e (b) regras de transformação referentes aos metamodelos das famílias de SISs, definidos anteriormente.

Para a especificação destes metamodelos arquiteturais relacionados a SISs, foram utilizadas em nossos cenários de uso, as famílias de sistemas de emergência pré-hospitalar e vigilância epidemiológica, discutidas anteriormente nesta dissertação. A partir da definição destes metamodelos foi possível especificar os cenários de uso, apresentados na Seção 4.2, para um sistema de emergência para vítimas de infarto do miocárdio pré-hospitalar (Correa et al., 2011) e um sistema de vigilância de doenças epidemiológicas (UK Biobank, 2012).

4.1.1 METAMODELAGEM PARA FAMÍLIA DE SISS PARA EMERGÊNCIA PRÉ-HOSPITALAR

A família de Sistemas de Emergência Pré-Hospitalar é ilustrada na Figura 4.1 sob a forma de um diagrama de classes UML, para melhor entendimento da metamodelagem. Nesse diagrama, os componentes arquiteturais "emergencista", "especialista" e "central de teleconsulta" foram modelados, respectivamente, nas classes "`Paramedic`", "`Specialist`" e "`Teleconsultation Center`". Esses componentes são descritos a seguir:

- O componente "`Paramedic`" é associado a "elementos de formulário" para dados de observação, e a "elementos de visualização" para dados de instrução, ilustrados respectivamente nas classes "`EHR Observation Form`" e "`EHR Instruction View`".
- O componente "`Specialist`" é associado a "elementos de visualização" para dados de observação, e a "elementos de formulário" para dados de instrução, ilustrados respectivamente nas classes "`EHR Observation View`" e "`EHR Instruction Form`".
- O componente "`Teleconsultation Center`" não é relacionado a GUI. Esse componente foi modelado para mostrar que os os componentes "`Paramedic`" e "`Specialist`" estão relacionados com uma central de teleconsulta

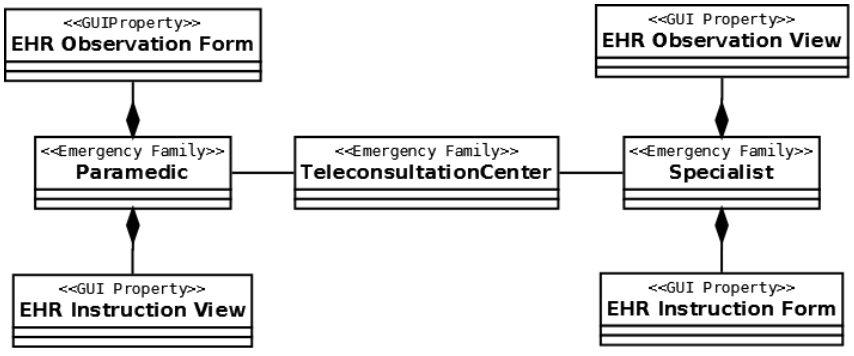


Figura 4.1: Principais componentes de uma Família de Sistemas de Emergência Pré-Hospitalar em diagrama de classes UML

O diagrama de sequência UML ilustrado na Figura 4.2 ilustra um aspecto específico de execução nos SISs nessa família, que é a bidirecionalidade de mensagens entre os componentes "`Paramedic`" e "`Specialist`": o primeiro envia dados de observação para o segundo, e este último envia dados de instrução para o primeiro.

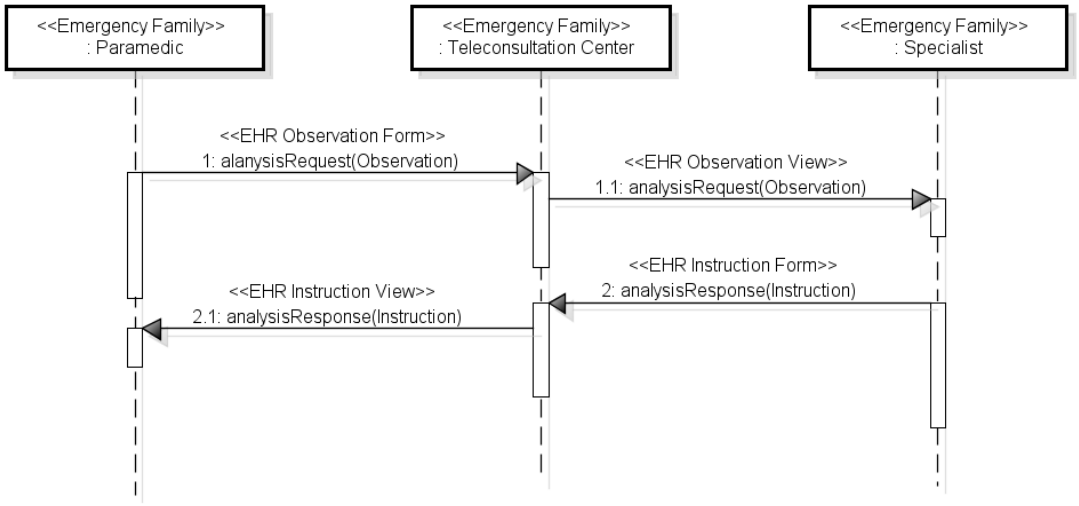


Figura 4.2: Principais componentes de uma Família de Sistemas de Emergência Pré-Hospitalar em diagrama de sequência UML

O metamodelo da Família de Sistemas de Emergência Pré-Hospitalar no formato Ecore é ilustrado na Figura 4.3. Como ilustrado, o pacote "`Emergency Family`" contém os elementos arquiteturais desta família de SISs. Este pacote contém o metadado "`Emergency System`", que permite instanciar os componentes arquiteturais referentes a esta família ("`TeleconsultationCenter`", "`Paramedic`" e "`Specialist`") em um sistema. Os componentes arquiteturais desta família foram definidos a partir do construtor "componente" ("`Component`") do metamodelo de Acme em Ecore apresentado na subseção 3.2.2. Esses componentes são associados a propriedades de GUI definidas no estilo arquitetural base apresentado na subseção 3.2.3.

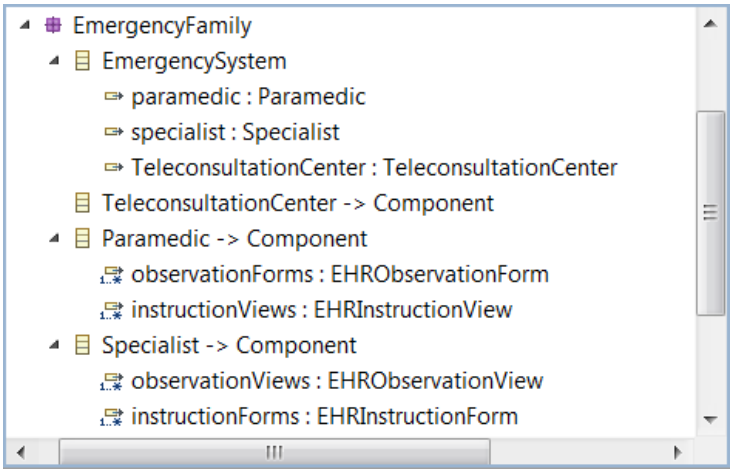


Figura 4.3: Metamodelo para a família de Emergência Pré-Hospitalar na especificação Ecore

4.1.2 METAMODELAGEM PARA FAMÍLIA DE SISS PARA VIGILÂNCIA EPIDEMIOLÓGICA

A Figura 4.4 ilustra a família de Sistemas de Vigilância Epidemiológica por meio de um diagrama de classes UML. Nesse diagrama, os componentes "coletor", "epidemiologista" e "central de dados" foram modelados, respectivamente, nas classes "`Collector`", "`Epidemiologist`" e "`Data Center`". Importante notar que, assim como na modelagem Ecore para a família de Emergência Pré-Hospitalar, apresentado anteriormente na subseção 4.1.1, os componentes arquiteturais da família de Sistemas de Vigilância Epidemiológica foram estendidos do construtor "componente" ("`Component`") a partir do metamodelo Acme no formato Ecore, para herdar suas características. Esses componentes são descritos a seguir:

- O componente "`Collector`" é associado a "elementos de formulário" para dados de observação, ilustrado na classe "`EHR Observation Form`".
- O componente "`Epidemiologist`" é associado a "elementos de visualização" para dados de observação, e a "elementos de formulário" para dados de avaliação, ilustrados respectivamente nas classes "`EHR Observation View`" e "`EHR Evaluation Form`".
- O componente "`Data Center`" não é relacionado a GUI. Esse componente foi modelado para mostrar que os os componentes "`Collector`" e "`Epidemiologist`" estão relacionados com uma central de dados.

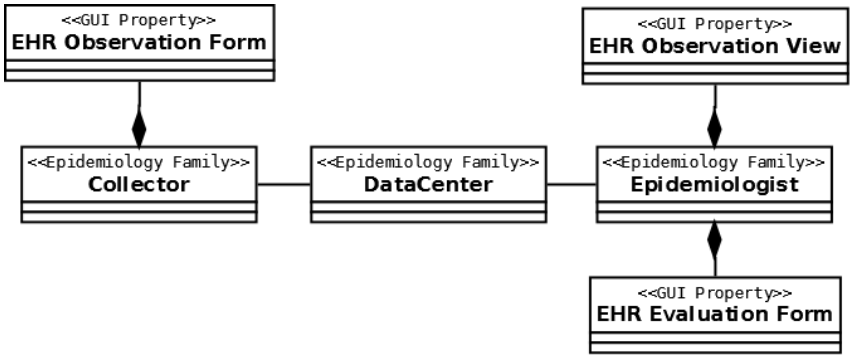


Figura 4.4: Principais componentes de uma Família de Sistemas de Vigilância Epidemiológica em diagrama de classes UML

O diagrama de sequência UML ilustrado na Figura 4.5 ilustra um aspecto específico de execução nos SISs nessa família, que é a unidirecionalidade de mensagens entre os

componentes "`Collector`" e "`Epidemiologist`": o primeiro componente envia dados de observação para o segundo, e este último apenas registra dados de avaliação.

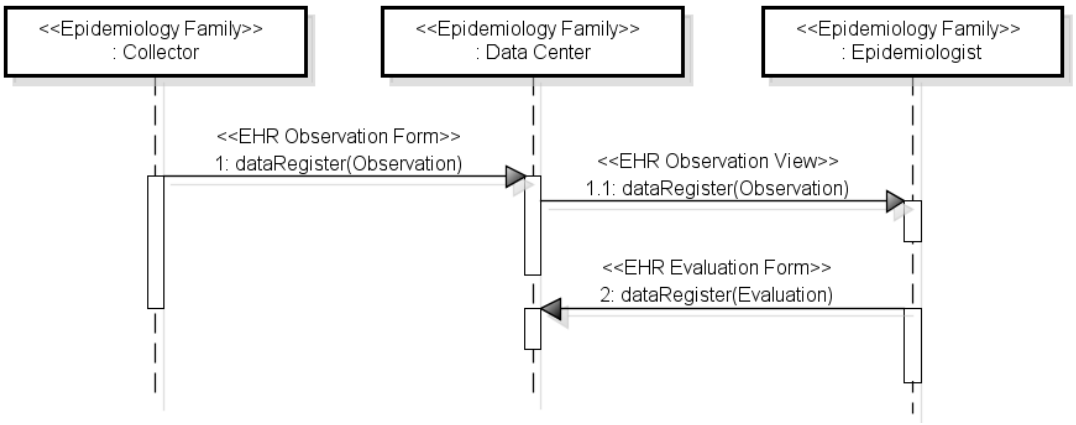


Figura 4.5: Principais componentes de uma Família de Sistemas de Vigilância Epidemiológica em diagrama de sequência UML

O metamodelo da Família de Sistemas de Vigilância Epidemiológica no formato Ecore é ilustrado na Figura 4.6. Esse metamodelo foi especificado de forma similar ao metamodelo de Emergência Pré-Hospitalar, com exceção das propriedades de GUI que foram associadas de forma particular aos componentes arquiteturais relacionados com GUI, além do metadado "`Epidemiology System`", que permite instanciar os componentes arquiteturais referentes a esta família em um sistema.

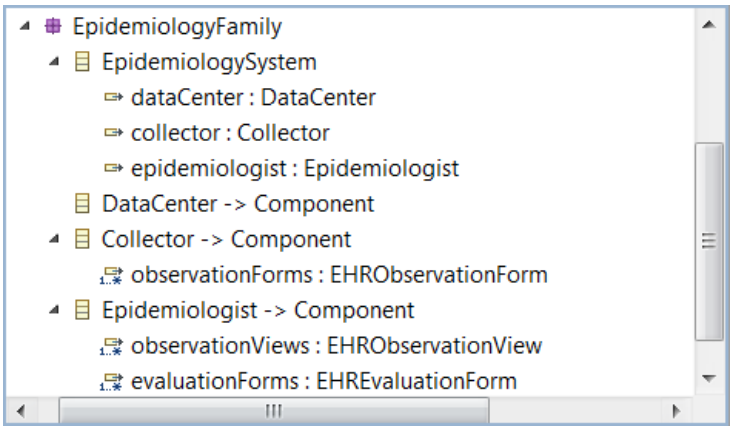


Figura 4.6: Metamodelo para a família de vigilância epidemiológica na especificação Ecore

4.1.3 REGRAS DE TRANSFORMAÇÕES PARA AS FAMÍLIAS DE SISS PARA EMERGÊNCIA PRÉ-HOSPITALAR E VIGILÂNCIA EPIDEMIOLÓGICA

O repositório de regras é constituído por três grupos de regras, conforme descrito no Capítulo 3. Esses grupos são executados em etapas para realizar as devidas transformações a partir dos metamodelos de origem e de destino. Mais especificamente, através de modelos arquiteturais que seguem metamodelos de famílias de SISs, como os metamodelos definidos nas subseções 4.1.1 e 4.1.2, o primeiro grupo de regras gera contêineres gráficos, como "portais", "documentos", entre outros, a partir dos componentes arquiteturais dessas famílias de SISs. Esses contêineres gráficos são preenchidos com componentes visuais de GUI específicos, como "legendas", "caixas de texto", "listas de seleção", entre outros, gerados por meio da atuação do terceiro grupo de regras sobre modelos de dados clínicos que seguem o metamodelo *open*EHR. O segundo grupo de regras funciona como um *binding* entre os outros dois grupos de regras, definindo entre outras coisas quais contêineres gráficos aninham quais componente visuais de GUI específicos.

A Figura 4.7 ilustra por meio de um diagrama de classes UML o primeiro e o segundo grupo de regras para as famílias de SISs para Emergência Pré-Hospitalar e Vigilância Epidemiológica (identificadas, respectivamente, pelos estereótipos "<<`emergency rule`>>" e "<<`epidemiology rule`>>"). Como ilustrado na Figura 4.5, o metadado "Portal", do metamodelo RichUbi (CIRILO et al., 2010), foi escolhido como alvo para as transformações de componentes arquiteturais de famílias de SISs para contêineres gráficos. Esse metadado permite separar de forma clara cada componente arquitetural destas famílias. Contudo, outros metadados desse metamodelo podem ser escolhidos para essas transformações, como "documento", "aba", ou até mesmo "div".

Os dois grupos de regras ilustrados na Figura 4.7 fornecem as informações arquiteturais necessárias ao grupo de regras de transformações de modelos de dados clínicos. É importante ressaltar que um dos benefícios da abordagem proposta neste trabalho é que o grupo de regras para transformação de modelos de dados clínicos é reusado entre diferentes famílias de SIS, pois é independente do grupo de regras de transformação para famílias de SISs utilizado. É importante notar também que os dois grupos de regras ilustrados na figura podem reutilizar (por herança) regras mais genéricas, referentes ao metamodelo da

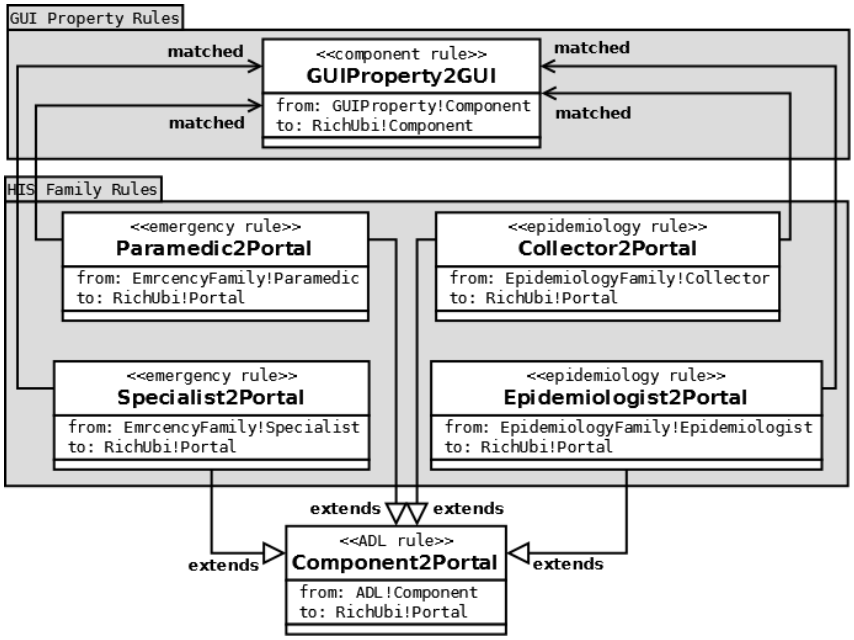


Figura 4.7: Grupos de regras responsáveis pelas famílias de SISs para Emergência Pré-Hospitalar e Vigilância Epidemiológica

linguagem Acme (identificadas pelo estereótipo "<<`ADL rule`>>").

4.2 MODELOS ECORE PARA OS CENÁRIOS DE USO

A segunda etapa na especificação dos cenários de uso utilizando o sistema de transformação IMT envolve o desenvolvedor de SISs, que tem o papel de definir modelos arquiteturais a partir dos metamodelos definidos pelo desenvolvedor de famílias (vide Seção 4.1), e associar esses modelos aos arquétipos *open*EHR que correspondem ao modelo de dados clínicos de interesse do sistema alvo. Nas subseções a seguir, as famílias de sistemas de emergência pré-hospitalar e de vigilância epidemiológica são usadas como base dos cenários de uso propostos, cenários estes que visam a modelagem arquitetural de dois sistemas pré-existentes.

4.2.1 CENÁRIO 1: MODELO ECORE DE UM SISTEMA PARA VÍTIMAS DE INFARTO

Neste primeiro cenário foi utilizado como sistema alvo um Sistema de Teleconsulta que permite a interação remota entre emergencistas e cardiologistas para vítimas de infarto agudo do miocárdio (CORREA et al., 2011), denominado AToMS (*Acute Myocardial*

Infarction Teleconsultation & Monitoring System). O AToMS foi desenvolvido no Laboratório Nacional de Computação Científica (LNCC) e validado junto a especialistas de domínio (médicos cardiologistas) da Universidade Federal do Rio de Janeiro (UFRJ).

O Infarto Agudo do Miocárdio (IAM) está entre uma das maiores causas de morte em todo o mundo (SMITH et al., 2011) e o risco de morte pode ser reduzido com tratamentos pré-hospitalares nos primeiros sintomas sofridos pelo paciente (KEELEY et al., 2003). Diante deste contexto, o AToMS permite que emergencistas realizem serviços de emergência pré-hospitalar a vítimas de IAM com apoio remoto de um cardiologista disponível em uma central de teleconsulta. Essa central, tipicamente, se localiza em ambientes hospitalares que possuem recursos adequados para o monitoramento continuo do paciente após sua remoção do local do primeiro atendimento, como uma unidade coronariana ou de tratamento intensivo. O AToMS tem o objetivo de aumentar as chances de sucesso de uma intervenção, principalmente em regiões remotas ou de difícil acesso.

A Figura 4.8 ilustra o diagrama de classes UML que representa o modelo arquitetural do AToMS baseado no metamodelo descrito na subseção 4.1.1, bem como os arquétipos necessários para este sistema específico. A Figura 4.8 representa emergencistas em ambulâncias ("`Ambulance Paramedic`") e médicos cardiologistas ("`Cardiologist`") interligados por uma central de teleconsulta. Emergencistas podem preencher formulários com dados clínicos de observação (identificados com o estereótipo "<<`observation form`>>"), e enviar essas informações ao médico cardiologista. Este último pode visualizar tais informações (identificadas com o estereótipo "<<`observation view`>>") e preencher formulários no sistema com dados de instrução (identificados com o estereótipo "<<`instruction form`>>"), que são enviados e visualizados (identificados com o estereótipo "<<`instruction view`>>") pelo emergencista. Para este cenário foram considerados arquétipos de "observação" validados por especialistas de domínio (CAVALINI et al., 2012) para uso no AToMS, como "frequência cardíaca" ("`Heart Rate`"), "pressão arterial" ("`Blood Pressure`") e "frequência respiratória" ("`Respirations`"). Arquétipos de instrução, tais como "ordem de medicação" ("`Medication Order`"), não tiveram seu uso validado por especialistas de domínio. Todos esses arquétipos podem ser encontrados no repositório *Clinical Knowledge Manager* (CKM, 2012) da fundação *open*EHR.

A Figura 4.9 ilustra o modelo Ecore para o AToMS. Vale ressaltar nessa figura como o desenvolvedor de SIS associa o arquétipo "pressão arterial" ("`openEHR EHR OBSERVA-`

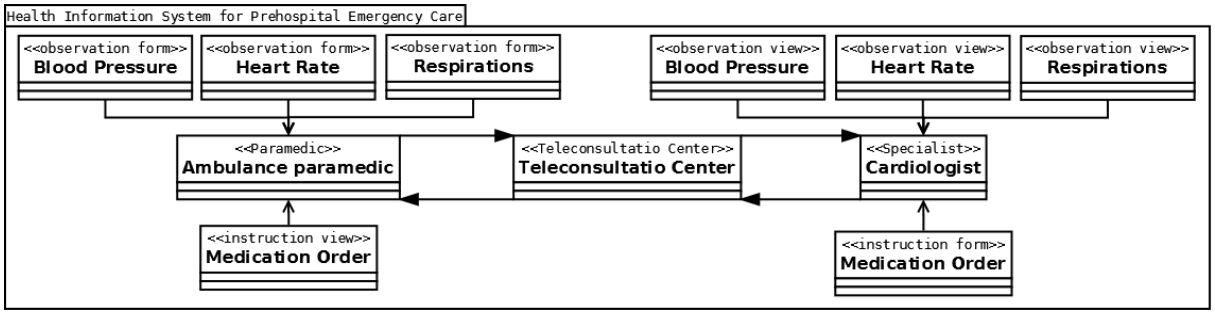


Figura 4.8: Modelo do sistema para vítimas de infarto.

`TION blood pressure v1`") ao modelo arquitetural do AToMS. Essa associação é feita na propriedade de GUI para formulários ("`EHR Observation Form Blood Pressure Paramedic Form`") do componente emergencista ("`Paramedic`").

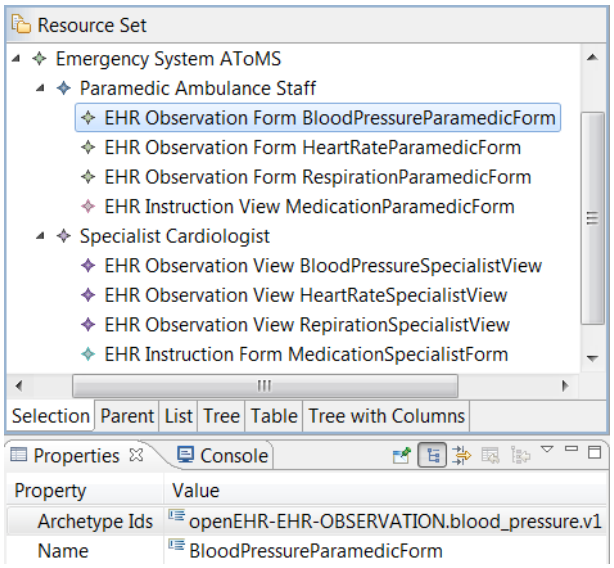


Figura 4.9: Modelo Ecore para o AToMS.

4.2.2 CENÁRIO 2: MODELO ECORE DE UM SISTEMA DE VIGILÂNCIA EPIDEMIOLÓGICA

O segundo cenário de uso utiliza como alvo um Sistema de Vigilância Epidemiológica chamado UK Biobank (UK Biobank, 2012).

UK Biobank é um sistema com o objetivo de coletar dados clínicos de centenas de milhares de pessoas. A organização detentora do UK Biobank seleciona e autoriza pesquisadores de domínio para a coleta de informações de dados clínicos com o objetivo

de disponibilizar essas informações para análise. Isso permite que outros pesquisadores de domínio se beneficiem dessas informações, como a identificação e avaliação de possíveis epidemias. Questionários para identificações epidemiológicas são elementos-chave no UK Biobank. Nesses questionários são encontradas perguntas do tipo "Seu pai biológico tem ou sofre de algum problema na saúde?", e o paciente pode responder com uma ou mais alternativas pré-definidas, como: "doença do coração"; "pressão alta"; "diabetes"; ou "nenhuma das anteriores". Também são coletadas nesse sistema medidas fisiológicas como altura, peso etc. Para utilizar esse sistema no cenário proposto, foram analisados e empregados arquétipos de observação e de avaliação presentes no repositório CKM da especificação *open*EHR, como descrito a seguir; contudo, é importante frisar que os modelos de dados clínicos empregados não foram validados por especialistas de domínio.

A Figura 4.10 ilustra o diagrama de classes UML que representa o modelo arquitetural do UK Biobank baseado no metamodelo descrito na subseção 4.1.2, bem como os arquétipos necessários para este sistema específico. Como ilustrado na Figura 4.10, os questionários do UK Biobank podem ser baseados no arquétipo "lista de verificação de saúde orientada" ("`A Health Oriented Check List`"), que pode ser especializado para a definição das perguntas e respostas de interesse. O código gerado para esses questionários para a visualização dos epidemiologistas deve ser estruturado com todas as informações coletadas periodicamente em forma de tabelas, gráficos, listas, etc. Arquétipos para medidas fisiológicas, como "altura/comprimento" ("`Height/Length`") e "peso corporal" ("`Body Weight`"), também foram usados neste cenário de uso.

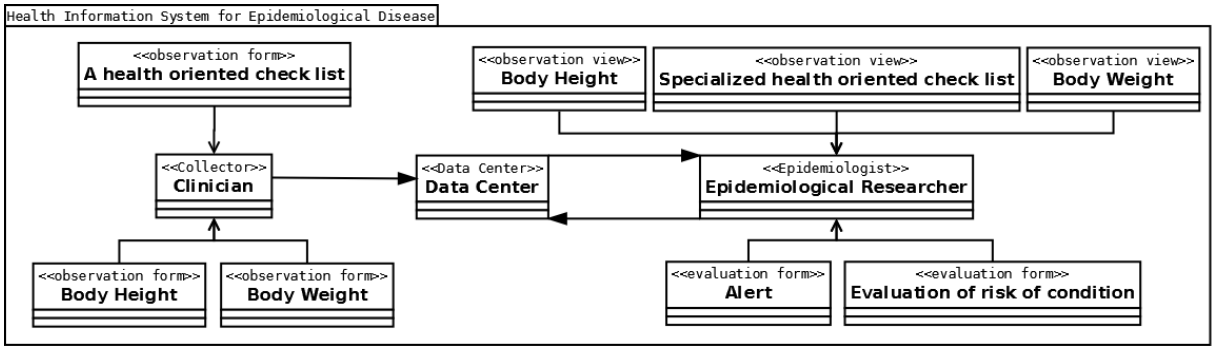


Figura 4.10: Modelo do sistema de vigilância epidemiológica.

A Figura 4.10 ainda ilustra que esses arquétipos podem ser utilizados pelos médicos coletores ("`Clinician`") e essas informações são enviadas ao centro de dados ("`Data Center`") para análise de um pesquisador em epidemiologia ("`Epidemiological Rese-`

`archer`"). O médico epidemiologista pode avaliar todos os dados enviados pelo médico coletor, e se julgar necessário, poderá registrar um possível alerta (como a iminência de uma epidemia). Como ilustrado na Figura 4.10, estes registros podem ser baseados em arquétipos de "avaliação", considerando a especificação *open*EHR, como "alerta" ("`Alert`") e "avaliação de risco de condição" ("`Evaluation of Risk of Condition`").

A Figura 4.11 ilustra o modelo Ecore para o UK Biobank. Vale ressaltar nessa figura como o desenvolvedor de SIS associa o arquétipo "avaliação de risco de condição" ("`openEHR EHR EVALUATION risk v1`") ao modelo arquitetural do UK Biobank. Essa associação é feita na propriedade de GUI para formulários (" `EHR Evaluation Form Epidemiologist Risk Form`") do componente epidemiologista (" `Researcher Epidemiologist`").

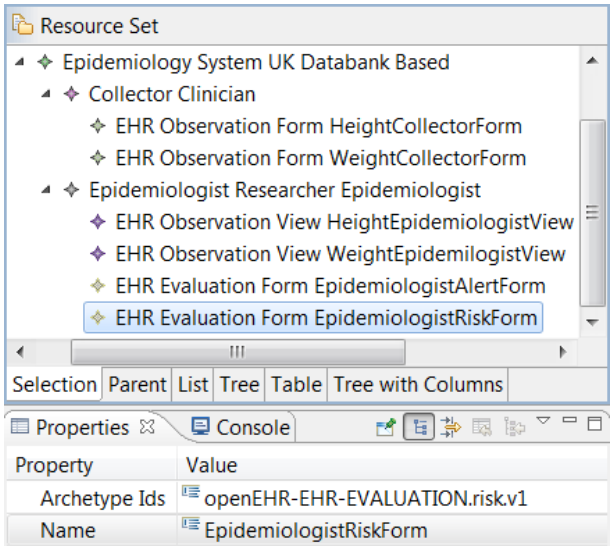


Figura 4.11: Modelo Ecore para o UK Biobank.

4.2.3 LIMITAÇÕES DE MODELAGEM

Distintos especialistas de domínio possuem necessidades distintas no uso de modelos de dados clínicos, considerando o repositório CKM do *open*EHR, para o desenvolvimento de SISs.

Uma limitação encontrada para o uso dos modelos de dados clínicos analisados anteriormente, é que alguns devem ser especializados para conter as informações de interesse no domínio. Assim, os elementos contidos nestes modelos de dados clínicos podem ser insuficientes ou desnecessários. Por exemplo, modelos de dados clínicos necessários para

o Sistema de Epidemiologia UK Biobank podem necessitar de especializações, como o arquétipo "lista de verificação de saúde orientada". Este arquétipo pode ser utilizado para a definição de questionários para fins epidemiológicos e deve ser especializado para a definição de perguntas e respostas pré-definidas. Outro exemplo é o arquétipo de instrução "ordem de medicação", que pode ser necessário para o AToMS. Esse arquétipo deve ser especializado para conter as devidas informações de instrução de domínio.

Outra limitação do sistema de transformação IMT é que o mesmo não utiliza um banco de dados para registrar e coletar as informações preenchidas nos modelos de dados clínicos pelos especialistas de domínio. Registrar as informações decorrentes de modelos de dados clínicos, com a utilização de um banco de dados, considerando a especificação *open*EHR, é uma frente de pesquisa complexa e está fora no escopo do trabalho proposto. Por exemplo, gerar código, de forma coerente, para as informações decorrentes dos modelos de dados clínicos para os questionários do sistema de epidemiologia UK Biobank requer o processamento de todas essas informações registradas pelo médico coletor. O resultado deste processamento deve ser analisado pelo epidemiologista considerando, por exemplo, tabelas, listas e gráficos. Modelos de dados clínicos como estes requerem uma análise dos dados armazenados para gerar códigos de GUI e está fora do escopo do trabalho proposto.

Considerando modelos de dados clínicos que devem ser especializados, como os arquétipos "lista de verificação de saúde orientada" e "ordem de medicação", discutidos anteriormente, tais especializações não foram encontradas na literatura para os cenários de uso propostos. Por esse motivo, os arquétipos supracitados não são tratados nos processos de transformação propostos no sistema de transformação IMT. Entretanto foi possível demonstrar a geração de código de GUI a partir de outros arquétipos, analisados anteriormente, como "altura/comprimento", "peso corporal", "alerta", "avaliação de risco de condição", "frequência cardíaca", "pressão arterial" e "frequência respiratória" referenciados em propriedades de GUI considerando informações arquiteturais de SISs. Isso foi possível porque a abordagem proposta pelo sistema de transformação IMT emprega metamodelagem e metaprogramação, o que permite transformações de modelos de dados clínicos de forma independente do seu conteúdo.

4.3 USO DO SISTEMA DE TRANSFORMAÇÃO IMT NOS CENÁRIOS DE USO

Esta seção apresenta a geração de código GUI para alguns modelos de dados clínicos (arquétipos) necessários para a especificação do AToMS (Cenário 1) e do UK Biobank (Cenário 2), ambos discutidos anteriormente.

O sistema de transformação IMT recebe como entrada os modelos de dados clínicos escolhidos e o modelo arquitetural de SIS pretendido e tem como resultado código de GUI com informações de domínio. A geração de código, considerando o sistema de transformação IMT, é realizada em duas etapas:

- A primeira etapa realiza processos de transformação M2M, considerando o repositório de regras de transformação. Essas regras foram desenvolvidas na linguagem ATL, o que permite a geração de modelos abstratos de GUI independentes de plataforma.
- A segunda etapa de transformações realiza processos de transformação M2C, considerando a ferramenta de geração de código GUI RichUbi (CIRILO et al., 2010).

4.3.1 TRANSFORMAÇÕES DE MODELOS PARA MODELOS (M2M)

A primeira etapa do processamento de transformações M2M tem como entrada os modelos de dados clínicos escolhidos e o modelo arquitetural do SIS pretendido e tem como resultado modelos abstratos de GUI.

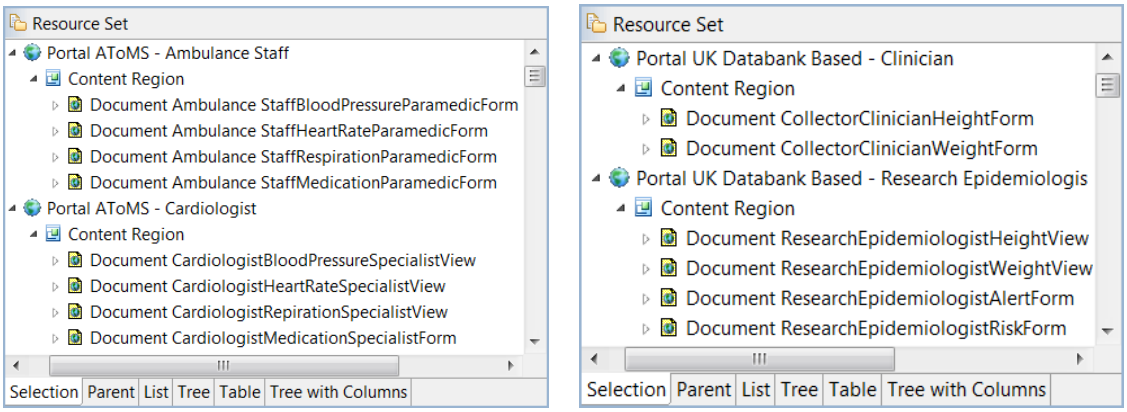
Conforme já dito, o repositório de regras é constituído por três grupos de regras para realizar as devidas transformações. De forma geral, o processamento dessas regras se dá da seguinte forma:

- Para cada componente arquitetural encontrado pelo primeiro grupo de regras no modelo arquitetural do SIS pretendido, esses componentes são transformados em portais web.
- O segundo grupo de regras realiza a ligação para o terceiro grupo de regras através das identificações de modelos de dados clínicos especificados nas propriedades de GUI. Por exemplo, arquétipos de observação “pressão arterial” e “avaliação de risco de condição”, são necessários, respectivamente, no AToMS e no UK Biobank.

Esses arquétipos são especificados em propriedades de GUI, respectivamente, como "formulários para observação" e "formulários para avaliação".

- Através da ligação estabelecida pelo segundo grupo de regras, o terceiro grupo de regras pode realizar as devidas transformações dos modelos de dados clínicos para "elementos de visualização" ou para "elementos de formulário". O código resultante destes modelos abstratos de GUI pode ser na forma de HTML, JSP e PHP, entre outros formatos.

Para realizar a geração de código de GUI a partir do sistema de transformação IMT, considerando os cenários 1 e 2, foram utilizados os arquétipos "altura/comprimento", "peso corporal", "alerta", "avaliação de risco de condição", "frequência cardíaca", "pressão arterial" e "frequência respiratória", encontrados no repositório CKM do *open*EHR. O resultado dos processos de transformação conduzidos pelos três grupos de regras é apresentado nas Figuras 4.12 e 4.13. A Figura 4.12 ilustra de forma sintética o resultado global dos processos de transformação de ambos os cenários. A Figura 4.13 detalha o resultado desses processos, em particular do terceiro grupo de regras, para o cenário 1 (AToMS).



(a) Sistema para vítimas de IAM. (b) Sistema para Epidemiologia UK Biobank

Figura 4.12: Modelos de GUI para SISs considerando o formato Ecore

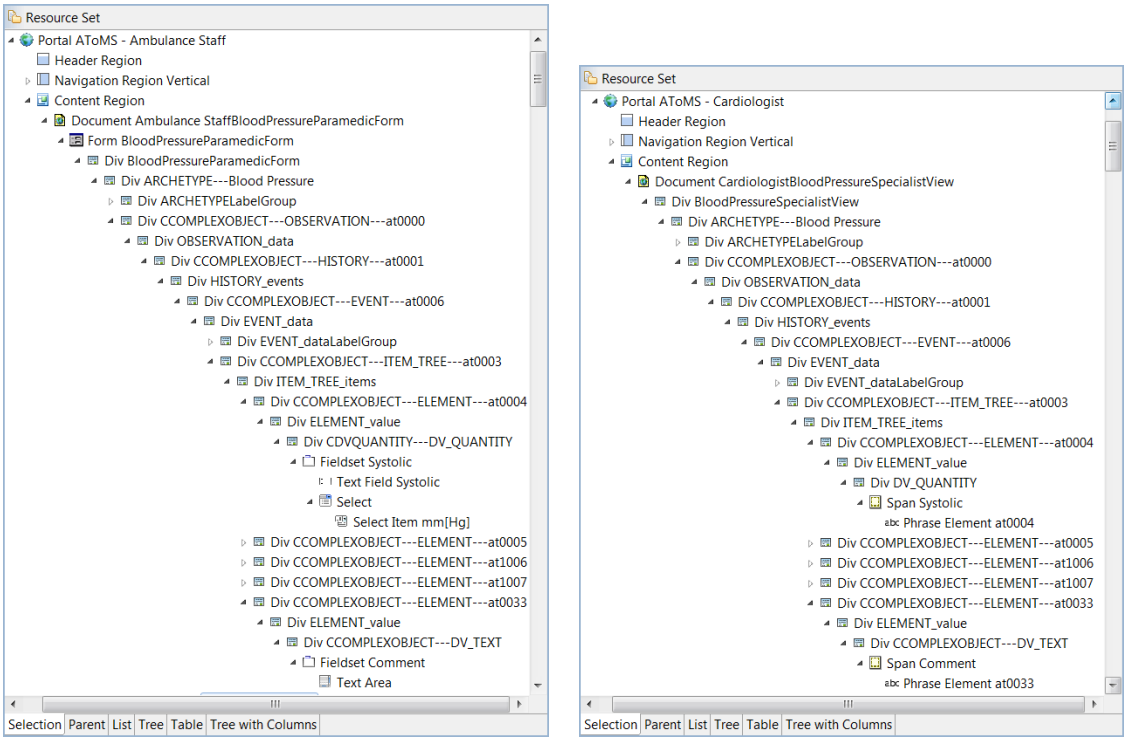
Os processos que resultaram nos modelos de GUI apresentados nas Figuras 4.12 e 4.13 são detalhados a seguir:

- O primeiro grupo transformou cada componente arquitetural dos sistemas considerados (AToMS e UK BioBank) em portais web, conforme apresentado na Figura 4.12, considerando o metadado "`Portal`" do metamodelo RichUbi. Para todos os portais

web, foram criados elementos para "região de conteúdo" ("`Content Region`"), que contêm documentos ("`Document`") para páginas web.

- O segundo grupo de regras transformou cada propriedade de GUI, definida nos componentes arquiteturais dos SISs, em documentos web (Figura 4.12, "`Document`"). Cada documento pode conter um ou mais modelos de dados clínicos abstratos de GUI. Por exemplo, como ilustrado na Figura 4.12(b), os arquétipos de avaliação "alerta" e "avaliação de risco de condição" foram transformados em documentos somente para o componente arquitetural "Epidemiologista" do UK BioBank. Os dois arquétipos supracitados são ilustrados na Figura 4.12, respectivamente, nos títulos dos documentos web "`Research Epidemiologist Alert Form`" e "`Research Epidemiologist Risk Form`".

- O terceiro grupo de regras pôde realizar as devidas transformações de modelos de dados clínicos, considerando as propriedades de GUI descritas nos componentes arquiteturais dos SISs. A Figura 4.13 ilustra as transformações para o arquétipo "pressão arterial" utilizado no modelo Ecore do AToMS apresentado na Seção 4.1.1, considerando os componentes arquiteturais emergencista de ambulâncias ("`Ambulance Staff`") e médico cardiologista ("`Cardiologist`"). A transformação deste arquétipo ocorre de forma distinta para esses componentes, devido às propriedades de GUI definidas, o que resulta, respectivamente, em elementos de formulários (Figura 4.13(a)) e elementos de visualizações (Figura 4.13(b)). A Figura 4.13 mostra ainda que o dado clínico "sistólico" (na identificação "`at0004`"), pertencente ao arquétipo "pressão arterial", foi transformado em elemento para formulário e visualização, ilustrado, respectivamente, nos elementos "`Field Set Sistolic`" e "`Span Systolic`", considerando o metamodelo RichUbi. É importante notar que a estrutura hierárquica das informações dos modelos de dados clínicos foi mantida. Isso foi possível devido a estruturação das regras de transformação, que segue a hierarquia dos metadados do metamodelo *open*EHR. Para manter essa estrutura hierárquica, foi utilizado o metadado "`div`" do metamodelo RichUbi nas regras de transformação.



(a) Elementos para formulário. (b) Elementos para visualização

Figura 4.13: Modelo de GUI gerado a partir do arquétipo “pressão arterial”, considerando distintos componentes do sistema AToMS.

4.3.2 TRANSFORMAÇÕES DE MODELOS PARA CÓDIGO (M2C)

A segunda etapa engloba as transformações M2C, que tem como entrada os modelos abstratos de GUI gerados pela etapa anterior e tem como resultado código de GUI. Nessa segunda etapa, o sistema de transformação IMT executa a ferramenta de geração de código de GUI RichUbi (CIRILO et al., 2010). As Figuras 4.14 e 4.15 ilustram uma síntese dos resultados desta etapa como uma consequência das regras de transformação de M2C da ferramenta RichUbi.

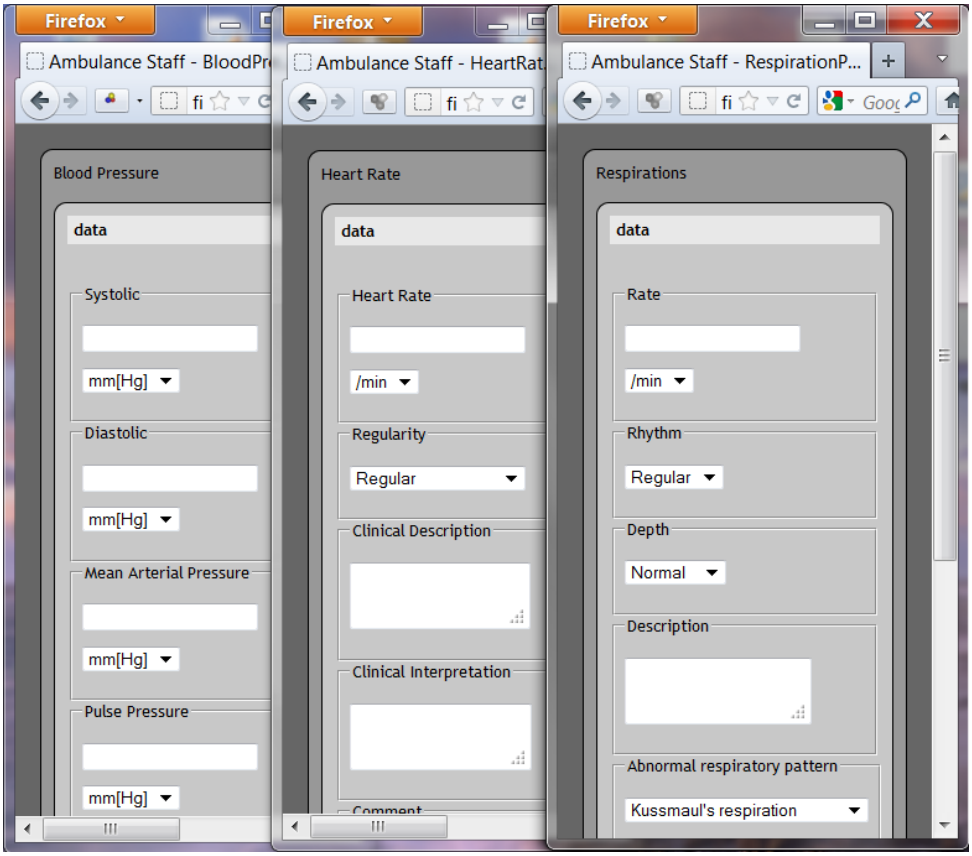
A Figura 4.14 apresenta a interface resultante dos arquétipos “pressão arterial” (“`Blood Pressure`”), “frequência cardíaca” (“`Heart Rate`”) e “frequência respiratória” (“`Respirations`”), que são necessários para o AToMS, considerando emergencistas de ambulâncias (“`Ambulance Staff`”) e o médico cardiologista (“`Cardiologist`”). Pode-se comparar a Figura 4.14 com a Figura 4.15, que apresenta uma síntese de outros arquétipos, como “peso corporal” (“`Body Weight`”), “altura/comprimento” (“`Height/Length`”), “alerta” (“`Alert`”) e “avaliação de risco de condição” (“`Evaluation of Risk of Condition`”), que são necessários para o UK Biobank. Foi gerado para cada modelo de dados clínicos, telas de GUI de acordo com o modelo arquitetural pretendido, considerando elementos de formulário e

visualização para os diferentes tipos de dados clínicos.

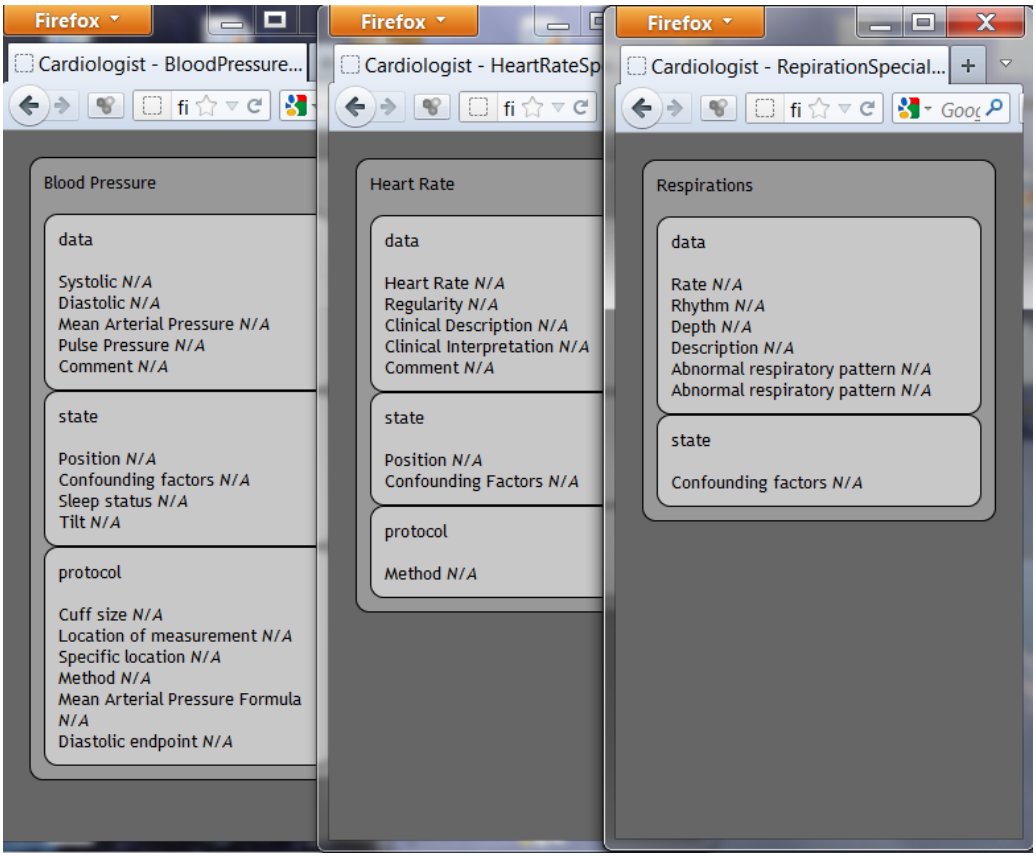
É importante observar que o código de GUI resultante dos processos de transformação de M2C, ilustrados nas Figuras 4.14 e 4.15, podem ser personalizados no nível de código, pois cada elemento destes modelos possui identificadores herdados em níveis mais abstratos, como os modelos gerados nos processos de transformação M2M. Por exemplo, os modelos ilustrados nas Figuras 4.14 e 4.15 foram personalizados com CSS (*Cascading Style Sheets*) (World Wide Web Consortium (W3C), 2012) para apresentar uma interface mais amigável, além de realçar as informações dos modelos de dados clínicos (arquétipos *open*EHR).

4.4 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Este capítulo detalhou o uso do sistema de transformação IMT na geração automática de código GUI para SISs a partir da síntese entre modelos de dados clínicos (arquétipos *open*EHR) desejados e o modelo de SIS pretendido, considerando dois cenários de uso caracterizados por duas famílias de SISs, considerando o estilo arquitetural de base para propriedades GUI proposto no Capítulo 3. Foram explorados que estilos arquiteturais, baseados em ADL, podem definir uma família de propriedades de GUI (estilo arquitetural base) que permita enriquecer famílias de SISs (estilo arquitetural de domínio). Essa abordagem foi realizada considerando o metametamodelo Ecore e transformações de M2M e M2C, com maior ênfase na definição de regras de transformação de M2M na linguagem ATL. Através destes cenários de uso foi possível gerar distintos códigos de GUI a partir de modelos de dados clínicos (do modelo *open*EHR) considerando informações arquiteturais enriquecidas com propriedades de GUI.

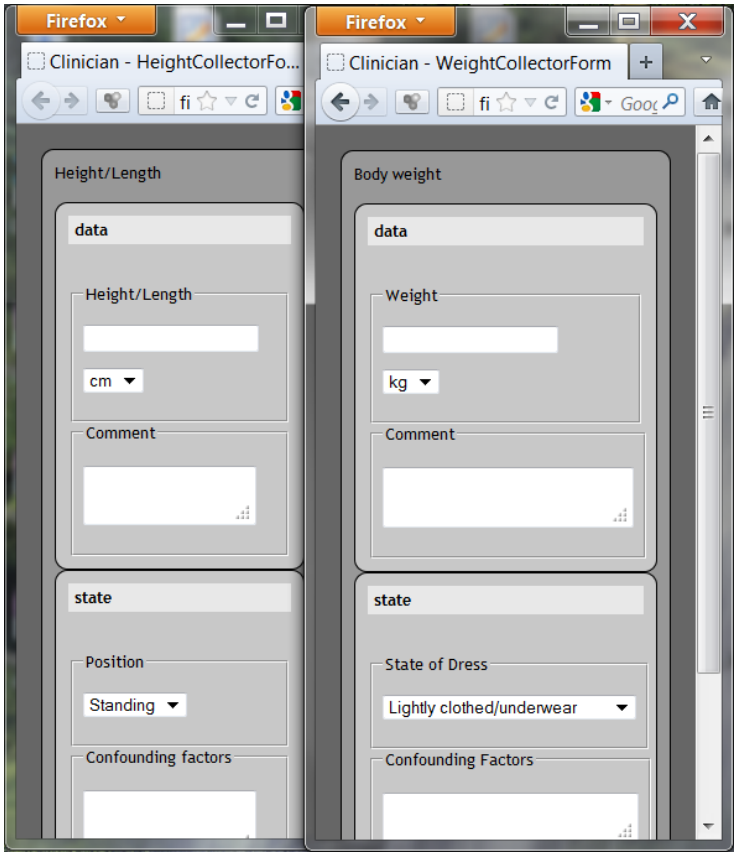


(a) Equipe de emergência

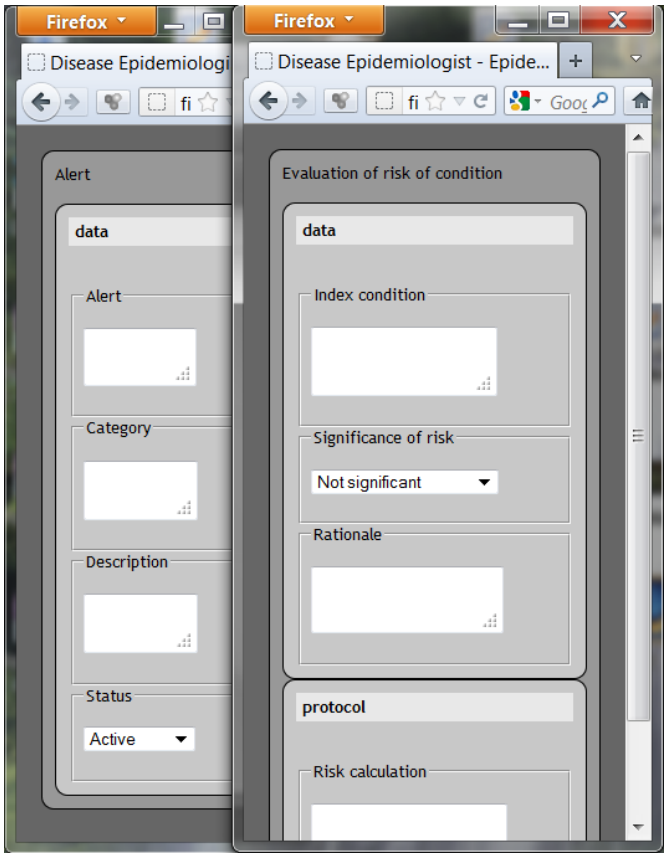


(b) Médico cardiologista

Figura 4.14: Interface resultante dos modelos de dados clínicos abstratos de GUI para o sistema AToMS.



(a) Médico coletor



(b) Pesquisador em epidemiologista

Figura 4.15: Código resultante dos modelos de dados clínicos abstratos de GUI para o sistema UK Biobank.

5 CONCLUSÕES

O uso de Tecnologias de Informação e Comunicação (TICs) na saúde ainda está em processo de consolidação. Contudo, tais tecnologias possuem potencial para auxiliar na aquisição, processamento e difusão da informação na saúde e podem favorecer demandas iminentes, ou em longo prazo, para práticas na saúde. Sistemas de Informação na Saúde (SISs) podem usufruir destes benefícios, contudo, ainda existem alguns fatores inibidores, de cunho tecnológico, que podem dificultar: (a) uma melhor interoperabilidade semântica entre distintos SISs (como hospitais, clínicas, laboratórios, etc.); (b) a redução do esforço no desenvolvimento de SISs; (c) a legibilidade em prontuários clínicos; e, o mais importante, (d) permitir a redução de erros médicos (CANTRILL, 2010). Diante destes fatores, esta dissertação utiliza especificações consolidadas na saúde e técnicas formais para a geração automática de código com informações de domínio. Este trabalho permite alcançar parcialmente os benefícios supracitados, por meio, por exemplo, da redução do esforço no desenvolvimento de SISs.

O fato da área de saúde ser descentralizada, considerando sua estrutura organizacional, mostra que as informações de um único indivíduo em SISs podem ser replicadas em diferentes contextos e distintas unidades na saúde (hospitais, clínicas, etc). Para reforçar a eficiência nos cuidados na saúde, tem sido explorada na literatura a adoção de Registros Eletrônicos de Saúde (RESs). Contudo o uso de RESs pode também se tornar um fator inibidor devido a sua complexidade de especificação e implementação.

Este trabalho apresentou uma proposta para tentar minimizar essa complexidade, que pode oferecer benefícios em algumas práticas médicas, considerando TICs, como Emergência Pré-Hospitalar (SASSER et al., 2005) e Vigilância Epidemiológica (CHEN et al., 2010). O trabalho proposto buscou mitigar essa complexidade através do uso de técnicas de DDM (Desenvolvimento Dirigido por Modelos) (STAHL; VÖLTER, 2006), além de apresentar na prática, o uso das especificações da organização *open*EHR (BEALE, 2002), como uma alternativa para RESs interoperáveis.

As especificações *open*EHR empregam modelagem multinível, o que facilitou, sobremaneira, a aplicação de técnicas de DDM ao longo deste trabalho. Dessa forma, essas técnicas de DDM podem mitigar fatores inibidores tecnológicos relacionados a manuten-

ção, evolução e independência de plataforma. A modelagem multinível *open*EHR permitiu automatizar processos de desenvolvimento de SISs com informações de domínio encontradas em modelos de dados clínicos, que podem ser construídos a partir do modelo de referência (RM - *Reference Model*) e modelo de arquétipos (AM - *Archetype Model*) das especificações *open*EHR. No contexto de DDM, um ou mais metamodelos permitem a construção de um ou mais modelos específicos com base em uma estrutura comum, considerando o domínio de metamodelos. Assim, os "modelos" RM e AM, à luz do presente trabalho, podem ser interpretados como metamodelos que permitem a construção particular de modelos de dados clínicos por especialistas na saúde. Esses modelos de dados clínicos podem facilitar o diálogo (ou melhor articulação) entre especialistas de domínio e desenvolvedores, com base nestes metamodelos. Para automatizar o desenvolvimento de SIS, foi considerado neste trabalho, o repositório de modelos de dados clínicos *Clinical Knowledge Manager* (CKM, 2012), pois esse repositório fornece modelos consolidados, ou em andamento, validados por especialistas de domínio. Modelos de dados clínicos podem ser construídos por especialistas de domínio, o que pode facilitar a transcrição do conhecimento em recursos computacionais.

5.1 CONTRIBUIÇÕES E SÍNTESE DOS PRINCIPAIS RESULTADOS

Conforme dito, a área da saúde é complexa e desafiadora para o desenvolvimento de SISs, considerando um maior nível de interoperabilidade, manutenibilidade e reuso. Neste contexto, modelos de dados clínicos podem facilitar o diálogo entre especialistas de domínio e desenvolvedores, porém esses modelos não possuem informações importantes para facilitar a geração de código, considerando, por exemplo, distintas arquiteturas de SISs. Em particular, a falta de informações arquiteturais pode elevar o custo em técnicas de DDM, considerando regras de Transformação de Modelos (CZARNECKI et al., 2005), que requerem conhecimento específico de linguagens de transformação, como linguagens baseadas em metamodelos.

Deste modo, este trabalho contribui para o avanço na geração de código para SISs, considerando o potencial de técnicas de DDM para a especificação e geração de aplicações na área de saúde. O trabalho apresentado indica ainda como aplicar técnicas de DDM, considerando regras de transformação de modelos, em metamodelos complexos, como os das especificações *open*EHR. Para mitigar potenciais custos de desenvolvimento,

foi proposta uma estratégia que pôde combinar modelos de dados clínicos e modelos arquiteturais, considerando metamodelagem em distintos níveis de abstração, com regras de transformação associadas a cada metamodelo. Essa estratégia, implica na utilização dos metamodelos *open*EHR (dados clínicos), além de metamodelos para famílias de SISs (arquitetura), como Emergência Pré-Hospitalar e Vigilância Epidemiológica, desenvolvidos segundo a construção de famílias de SISs (BRAGA et al., 2012) e (GOMES et al., 2012). Os metamodelos desenvolvidos neste trabalho, referentes a essas famílias, podem especificar sistemas particulares, considerando reuso das principais estruturas (arquiteturais semelhantes) de domínio definidas nessas famílias de SISs. A definição da estratégia proposta permite que o desenvolvedor especifique metamodelos e regras de transformações para famílias de SISs. Assim, foi possível gerar código para SISs a partir de modelos de dados clínicos de interesse, considerando arquiteturas de sistemas particulares. Regras de transformação podem ser agrupadas para cada metamodelo envolvido, portanto, o resultado deste trabalho pode contribuir no melhor reuso de regras de transformação, responsáveis pela tradução de modelos de dados clínicos associados a modelos arquiteturais.

SISs completos possuem vários elementos de software, como tecnologias para comunicação, persistência de dados, e, principalmente, interface gráfica de usuário (GUI - *Graphical User Interface*). Este último, particularmente, é um dos elementos mais importantes em SISs, pois, tipicamente, esse elemento é o primeiro contato dos usuários finais (especialistas da saúde) com o sistema. O trabalho proposto explorou o projeto de GUI, o qual pode facilitar o diálogo entre especialistas de domínio e desenvolvedores de software. O trabalho apresentado propõe o sistema de transformação IMT, que emprega a estratégia detalhada no parágrafo anterior, para a síntese automática de GUI para SISs, considerando modelos de dados clínicos e modelos arquiteturais. Além do uso de especificações de dados clínicos consolidadas, como *open*EHR, foi empregado na modelagem arquitetural algumas construções inspiradas em Linguagens de Descrição Arquitetural (*Architecture Description Languages* - ADLs), como Acme (GARLAN et al., 1997). O sistema de transformação IMT especifica um estilo arquitetural base que permite aos modeladores de SISs enriquecer famílias de SIS com propriedades fundamentais associadas a GUIs, como “Elemento de visualização” e “Elemento de formulário”, por exemplo. Considerando que a estratégia consiste em definir grupos de regras de transformação para cada metamo-

delo, a família de propriedades também possui um grupo de regras associadas. Esse estilo arquitetural base permitiu o desenvolvimento, de forma particular, de famílias de SISs, como Emergência Pré-Hospitalar e Vigilância Epidemiológica. Desse modo as regras de transformação referentes ao metamodelo de propriedades de GUI e aos metamodelos de dados clínicos (*open*EHR) foram reutilizadas. Através de padrões de projeto, como *Composite* (GAMMA et al., 1995), foi possível obter maior reuso das regras desenvolvidas, além de simplificar, elaborar e aplicar as regras de transformação desenvolvidas.

5.2 LIMITAÇÕES

O uso de estilos arquiteturais para definir metamodelos como especificações de famílias genéricas para a especialização de famílias de domínios pode oferecer vantagens para técnicas de DDM, como a definição de regras de transformação para qualquer modelo de dado clínico de forma independente de distintas especificações arquiteturais. Entretanto, o emprego de regras de transformação baseadas em metamodelos (como proposto pelo OMG) impõe algumas limitações, como o fato de que a criação ou modificação de elementos nos metamodelos de SISs (famílias de domínio), bem como de relacionamentos entre estes elementos, implica na criação ou modificação de regras (e relacionamentos entre regras) correspondentes. Essas limitações podem ter um alto custo de desenvolvimento para o *desenvolvedor de famílias*. Contudo, esse esforço pode ser reduzido a partir do reuso de regras de transformação para os metamodelos de dados clínicos. A redução desse esforço pode ser alcançada através das especificações propostas no sistema de transformação IMT, como a modelagem de famílias de SISs, considerando um metamodelo genérico para propriedades de GUI (um estilo arquitetural base para a definição de uma família genérica). As regras de transformação relacionadas a esse metamodelo genérico permitem a ligação entre as regras dos metamodelos de SISs e as dos metamodelos de dados clínicos.

Outra questão é que o sistema de transformação IMT não gera código de GUI relacionado com persistência em banco de dados, considerando elementos de formulários e elementos de visualização, a partir de modelos de dados clínicos. Essa funcionalidade é prevista a partir da integração do sistema de transformação IMT a uma linha de produto de software, como discutido na seção seguinte.

5.3 TRABALHOS FUTUROS

O sistema de transformação IMT permite a geração de código automático para SISs através de técnicas de DDM, considerando uma estratégia na associação entre modelos de dados clínicos e informações arquiteturais. A estratégia permite que o desenvolvedor defina informações arquiteturais, através de famílias de SISs e regras de transformação relacionadas a essas famílias. Essa estratégia permite melhor reuso das regras de transformação na geração de código a partir de modelos de dados clínicos. Apesar do sistema de transformação proposto demonstrar benefícios tecnológicos potenciais, como explorado nas provas de conceito para os cenários de uso em sistemas de emergência pré-hospitalar e vigilância na saúde, é necessário aplicar o sistema proposto em casos reais. Assim, será possível tentar se alcançar, de fato, o principal objetivo dos SISs, que é aumentar a eficácia do cuidado em saúde por meio de uma melhor comunicação entre sistemas, uma maior legibilidade em prontuários clínicos e, consequentemente, uma redução de erros médicos.

Como trabalho futuro, o sistema de transformação IMT pode ser estendido para facilitar o desenvolvimento de regras de transformação para a geração de código para SISs de forma mais intuitiva sem a necessidade do uso de linguagens específicas de transformação de modelos, como ATL. Pretende-se com esta extensão realizar composição de modelos, com o objetivo de gerar automaticamente regras de transformação. Uma possível forma de alcançar esse objetivo consiste no uso de tecnologias como AMW (*Atlas Model Weaving*) (FABRO et al., 2005). Com essa tecnologia, o desenvolvedor pode utilizar, por exemplo, aplicativos que permitem a especificação de famílias de SISs e associações, de forma gráfica (p.ex. *links*), entre modelos de dados clínicos e informações arquiteturais de interesse. Com essas associações é possível gerar automaticamente regras de transformação para a geração de código de GUI.

Pode-se também estender o sistema de transformação desenvolvido para facilitar a geração de código utilizando outros construtores de ADLs para definição de estilos arquiteturais como conectores, portas, restrições, *bindings*, entre outros. Esses construtores podem permitir geração de código para outros elementos de software de SISs, como modelos para persistência em bancos de dados ou modelos para troca de mensagens entre componentes de um SIS. Para esse fim, é possível empregar novamente estilos arquiteturais base, como uma família de propriedades genérica para SISs, como uma forma de minimizar o esforço de desenvolvimento. Essas extensões podem ser desenvolvidas no âm-

bito do projeto SPLiCE (*Software-Product Lines in HealthCarE*) (GOMES et al., 2012), desenvolvido no INCT-MACC [1] (Instituto Nacional de Ciência e Tecnologia em Medicina Assistida por Computação Científica).

Outra questão que pode ser explorada no sistema de transformação IMT, no nível das transformações M2C, é a adoção de padrões de software como *Model-View-Controller* (MVC), que permite a modularização entre "modelo", "visualização" e "controle" para persistência de dados. Considerando o padrão MVC, o sistema de transformação IMT permite, somente, a geração de código para a "visualização" de GUI através de "modelos" de dados clínicos e "modelos" arquiteturais, para a definição de famílias de SISs. O padrão MVC pode facilitar a geração de código para o "controle"de persistência de dados para esses sistemas. Uma possível estratégia para essa integração é a definição de um metamodelo abstrato para o padrão MVC, e as instâncias (modelos resultantes) podem ser associadas com as propriedades dos modelos de SISs, para a especificação de "controles". Nesta estratégia, é possível definir regras de transformação para a geração de código para os elementos de SISs com propriedades para o controle do sistema.

[1] http://macc.lncc.br

Apêndice A - REGRAS DE TRANSFORMAÇÃO

A descrição completa das regras de transformação, apresentadas ao longo deste trabalho de forma esquemática (diagramas UML), é feita neste apêndice utilizando a sintaxe concreta da linguagem ATL, com o objetivo de complementar a argumentação proposta.

As regras foram desenvolvidas na plataforma *Eclipse Modeling Framework* (EMF) (Eclipse Foundation, 2012), considerando o metametamodelo Ecore, desta mesma plataforma. É importante lembrar que o metametamodelo Ecore é a base de todos os metamodelos envolvidos neste trabalho. Antes de demonstrar as regras, é necessário apresentar o módulo (ou cabeçalho) das regras de transformação ATL desenvolvidas. Esse módulo permite a especificação de variáveis para regras de transformação, considerando modelos e metamodelos para origem e destino. Além disso, foram desenvolvidas funções auxiliares (*helpers* da linguagem ATL) para obter mais informações dos modelos de dados clínicos, como a terminologia referente ao código de um determinado elemento.

MÓDULO DE TRANSFORMAÇÃO

```
module OpenEHR2RichUbi;
create richUbiM : richUbiMM from archM : archMM, openEHRM:
   openEHRMM;
uses LibHelpers;
```

Como apresentado no código ATL acima, os elementos deste módulo são definidos por modelos e metamodelos de origem (`from`) e de destino (`create`):

- `openEHRMM`: representa o metamodelo *open*EHR.
- `openEHRM`: representa uma variável para um único modelo com todos os modelos de dados clínicos de interesse, conforme descrito na Seção 3.2.1. `openEHRM` é uma especificação de `openEHRMM`.
- `archMM`: representa o metamodelo para famílias de SISs, como Emergência Pré-Hospitalar e Vigilância epidemiológica, sendo estas enriquecidas, de forma particular, com elementos do metamodelo da família de propriedades de GUI.

- `archM`: representa uma variável para um modelo arquitetural de SIS. `archM` é uma especificação de `archMM`.
- `richUbiMM`: representa o metamodelo de GUI RichUbi.
- `richUbiM`: representa uma variável para o modelo de GUI a ser gerado pelas regras de transformação. `richUbiM` é uma especificação de `richUbiMM`.
- `LibHelpers`: é uma biblioteca de funções auxiliares para capturar informações extras de `openEHRM`. Essas funções são utilizadas em algumas regras de transformação.

REGRAS DE TRANSFORMAÇÃO ATL

A partir do conhecimento da estrutura hierárquica dos metamodelos de origem e destino, o trabalho apresentado propõe a definição de grupos de regras de transformação, de forma estruturada, para cada metamodelo envolvido. As regras propostas foram estruturadas em três grupos principais:

- As regras do primeiro grupo estão associadas aos metamodelos de famílias de SISs;
- As regras do segundo grupo estão associadas ao metamodelo da família de propriedades de GUI.
- Por fim, as regras do terceiro grupo estão associadas ao metamodelo de dados clínicos (*open*EHR);

Outras regras de transformação, referentes ao metamodelo Acme no formato Ecore, podem ser reutilizadas no primeiro grupo de regras de transformação referentes aos componentes de famílias de SISs, como apresentado no código ATL abaixo:

```
-- Regra declarativa referente ao metamodelo Acme
abstract rule Component2Portal{
    from
        comp: archMM!Component
    to
        richUbiM: richUbiMM!Portal (
            name <- comp.refImmediateComposite().name +
                ' - ' + comp.name,
            documentsExtension <- 'html', -- esse parâmetro pode
               ser capturado externamente
            headerRegion <- hrVar,
```

```
            navigationRegion <- nrVar,
            footerRegion <- frVar
        ),
        hrVar : richUbiMM!HeaderRegion(

        ),
        nrVar: richUbiMM!NavigationRegion(
            linkGroups <- linkGroup
        ),
        frVar : richUbiMM!FooterRegion(

        ),
        linkGroup: richUbiMM!LinkGroup(
            title <- comp.name
        )
}
```

O primeiro grupo de regras de transformação está relacionado aos componentes das Famílias de SISs para Emergência Pré-Hospitalar e Vigilância Epidemiológica. O objetivo destas regras é gerar contêineres gráficos, como "portais". Essas regras podem ser observadas no código ATL abaixo:

```
-- Primeiro grupo de regras declarativas referente aos metamodelos
    de famílias de SISs (HIS Family rules)

-- Regras para a Familia de Emergencia Pre-Hospitalar

rule Paramedic2Portal extends Component2Portal{
    from
        comp:archMM!Paramedic
    to
        richUbiM: richUbiMM!Portal(
            contentRegion <- crVar
        ),
        crVar: richUbiMM!ContentRegion(
            documents <- comp.observationForms,
            documents <- comp.instructionViews
    )
}

rule Specialist2Portal extends Component2Portal{
    from
        comp:archMM!Specialist
    to
        richUbiM: richUbiMM!Portal(
            contentRegion <- crVar
        ),
        crVar : richUbiMM!ContentRegion(
            documents <- comp.observationViews,
```

```
            documents <- comp.instructionForms
        )
}

-- Regras para a Familia de Vigilancia Epidemiológica --

rule Collector2Portal extends Component2Portal{
    from
        comp: archMM!Collector
    to
        richUbiM: richUbiMM!Portal(
            contentRegion <- crVar
        ),
        crVar: richUbiMM!ContentRegion(
            documents <- comp.observationForms
        )
}

rule Epidemiologist2Portal extends Component2Portal{
    from
         comp: archMM!Epidemiologist
    to
        richUbiM: richUbiMM!Portal(
            contentRegion <- crVar
        ),
        crVar: richUbiMM!ContentRegion(
            documents <- comp.observationViews,
            documents <- comp.evaluationForms
    )
}
```

O segundo grupo de regras funciona como um *binding* (ligação) entre os outros dois grupos de regras, definindo entre outras coisas quais contêineres gráficos aninham quais componente visuais de GUI específicos. O código a seguir apresenta esse grupo de regras, onde cada regra possui um metadado de origem (referente ao metamodelo de propriedades de GUI), e um ou mais metadados de destino (referentes ao metamodelo RichUbi):

```
-- Segundo grupo de regras declarativas referente ao metamodelo de
    propriedades de GUI (GUI Property Rules)

abstract rule PropertyEHR2Document{
    from
        guiComp: archMM!PropertyEHR
    to
        doc: richUbiMM!Document(
            fileName <- guiComp.refImmediateComposite().name
                    + guiComp.name,
            title <- guiComp.refImmediateComposite().name
```

```
                    + ' - ' + guiComp.name
        ),
        divPropertyEHR: richUbiMM!Div(
            id <- guiComp.name,
            blockLevelComponents <- guiComp.archetypeIds ->
                        collect(e|thisModule.ARCHETYPE2Div
                            (guiComp.getArchetype(e),guiComp))
        )
}

-- EHRForm2Form: Para toda propriedade EHRForm é gerado um
   Document
rule EHRForm2Form extends PropertyEHR2Document{
    from
        guiComp: archMM!EHRForm
    to
        doc: richUbiMM!Document(
            forms <- formVar
        ),
        formVar: richUbiMM!Form(
            name <- guiComp.name,
            blockLevelComponents <- divPropertyEHR)
}

-- EHRForm2Form: Para toda propriedade EHRView é gerado um
   Document
rule EHRView2View extends PropertyEHR2Document{
    from
        guiComp: archMM!EHRView
    to
        doc: richUbiMM!Document(
            blockLevelComponents <- divPropertyEHR
        )
}
```

O terceiro grupo de regras (para modelos de dados clínicos) é responsável pelas transformações dos modelos de dados clínicos e seus respectivos elementos de acordo com as propriedades de GUI. Essas transformações consistem em reproduzir elementos de formulários (como os metadados para "*list box*" e "*text area*" do metamodelo RichUbi) ou elementos de visualização (como o metadado para "*span*" do metamodelo RichUbi). O código abaixo apresenta as regras de transformação, onde cada regra possui, no mínimo, dois metadados de origem (o tipo de propriedades de GUI, o tipo de dados clínico e um terceiro para capturar mais informações dos arquétipos). Por outro lado, essas regras possuem um ou mais metadados de destino (como o tipo de GUI):

```
-- Terceiro grupo de regras declarativas referente ao metamodelo
   openERH (openEHR Rules)

lazy rule ARCHETYPE2Div{
    from archetype: openEHRMM!ARCHETYPE,
            guiComp: archMM!PropertyEHR
    to d: richUbiMM!Div(
        id <- archetype.oclType().name + '---' +
                thisModule.getOntName(archetype.concept,archetype)
                    ,
            class <- archetype.oclType().name,
        blockLevelComponents <- thisModule.createSpan
        (archetype.oclType().name, thisModule.getOntName
                ('at0000' ,archetype)),
        blockLevelComponents <- thisModule.COBJECT2Div
            (archetype.definition, guiComp,archetype)
    )
}

lazy rule COBJECT2Div{
    from
        co: openEHRMM!COBJECT,
            guiComp: archMM!PropertyEHR,
                archetype: openEHRMM!ARCHETYPE
    to
        divVar: richUbiMM!Div (
            id <- co.oclType().name
                +'---'+ co.rmTypeName +'---'+ co.nodeId
        )
}

lazy rule CCOMPLEXOBJECT2Div extends COBJECT2Div{
    from
        co: openEHRMM!CCOMPLEXOBJECT,
            archetype: openEHRMM!ARCHETYPE
    to
        divVar: richUbiMM!Div (
            blockLevelComponents <- thisModule.createSpan
                (co.rmTypeName, co.rmTypeName),
            blockLevelComponents <- co.attributes ->
                collect(e | thisModule.CATTRIBUTE2Div
                    (e,guiComp,archetype))
        )
}

lazy rule CATTRIBUTE2Div{
    from
        cattribute : openEHRMM!CATTRIBUTE,
            guiComp: archMM!PropertyEHR,
                archetype: openEHRMM!ARCHETYPE
    to
```

```
        divVar : richUbiMM!Div(
            id <- cattribute.refImmediateComposite().rmTypeName
                + '_'  + cattribute.rmAttributeName,
            class <- attribute.refImmediateComposite().rmTypeName
                + '_'  + cattribute.rmAttributeName,
            blockLevelComponents <- thisModule.createSpan
                (divVar.id,cattribute.rmAttributeName),
            blockLevelComponents <- cattribute.children ->
                collect(e | thisModule.COBJECT2Div
                    (e,guiComp,archetype))
        )
}

lazy rule CCOMPLEXOBJECT_DVTEXT2Form extends COBJECT2Div{
    from
        co: openEHRMM!CCOMPLEXOBJECT,
            guiComp: archMM!EHRForm,
                archetype: openEHRMM!ARCHETYPE
                    (co.rmTypeName='DV_TEXT')
    to
        divVar : richUbiMM!Div(
            id <- co.oclType().name +'---'+ co.rmTypeName,
            blockLevelComponents <- thisModule.createSpan
                (co.rmTypeName,co.rmTypeName),
            fieldsets <- fieldsetVar
        ),
        fieldsetVar : richUbiMM!Fieldset(
            legend <- thisModule.getOntName
                (co.refImmediateComposite().
                    refImmediateComposite().nodeId,archetype),
            controls <- textAreaVar
        ),
        textAreaVar : richUbiMM!TextArea(
             value <- ''  -- valor a ser inserido pelo usuario
        )
}

lazy rule CCOMPLEXOBJECT_DVTEXT2View extends COBJECT2Div{
    from
        co: openEHRMM!CCOMPLEXOBJECT,
            guiComp: archMM!EHRView,
                archetype: openEHRMM!ARCHETYPE
                    (co.rmTypeName='DV_TEXT')
    to
        divVar : richUbiMM!Div(
            id <- co.oclType().name+'---'+ co.rmTypeName,
            blockLevelComponents <- thisModule.createSpan
                (co.rmTypeName,co.rmTypeName),
            inlineComponents  <- spanVar
        ),
        spanVar : richUbiMM!Span(
```

```
            text <-   thisModule.getOntName
                (co.refImmediateComposite().
                    refImmediateComposite().nodeId,archetype),
            inlineComponents <- phraseVar_value
        ),
        phraseVar_value : richUbiMM!PhraseElement(
            text <- 'N/A',
            id <- co.refImmediateComposite().
                    refImmediateComposite().nodeId
        )
}

lazy rule CCOMPLEXOBJECT_DV_CODED_TEXT2Form extends COBJECT2Div{
    from
        co: openEHRMM!CCOMPLEXOBJECT,
            guiComp: archMM!EHRForm,
                archetype: openEHRMM!ARCHETYPE
                    (co.rmTypeName='DV_CODED_TEXT')
    to
        divVar : richUbiMM!Div(
            id <- co.oclType().name
                +'---'+ co.rmTypeName,
            blockLevelComponents <- thisModule.createSpan
                (co.rmTypeName,co.rmTypeName),
            fieldsets <- fieldsetVar
        ),
        fieldsetVar : richUbiMM!Fieldset(
            legend <- thisModule.getOntName
                (co.refImmediateComposite().
                    refImmediateComposite()
                        .nodeId,archetype),
            controls <- textAreaVar
        ),
        textAreaVar : richUbiMM!Select(
            id <- guiComp.refImmediateComposite().name
                +'---'+archetype.archetypeId.value
                    + '---' +co.refImmediateComposite().
                        refImmediateComposite().nodeId,
            items <- co.attributes -> collect(e | e)->first().
                children -> collect(e | e)->first().
                    codeList -> collect(e | thisModule.
                        CreateSelectItem
                            (e,archetype))
        )
}

lazy rule CCOMPLEXOBJECT_DV_CODED_TEXT2View extends COBJECT2Div{
    from
        co: openEHRMM!CCOMPLEXOBJECT,
            guiComp: archMM!EHRView,
                archetype: openEHRMM!ARCHETYPE
```

```
                    (co.rmTypeName='DV_CODED_TEXT')
    to
        divVar : richUbiMM!Div(
            id <- co.oclType().name
                +'---'+ co.rmTypeName,
            blockLevelComponents <- thisModule.
                createSpan(co.rmTypeName,co.rmTypeName),
            inlineComponents  <- spanVar
        ),
        spanVar : richUbiMM!Span(
            text <-   thisModule.
                getOntName(co.refImmediateComposite().
                    refImmediateComposite().
                        nodeId,archetype),
            inlineComponents <- phraseVar_value
        ),
        phraseVar_value : richUbiMM!PhraseElement(
            text <- 'N/A', -- valor a ser resgatado em BD
            id <- co.refImmediateComposite().
                refImmediateComposite().nodeId
        )
}

lazy rule CDVQUANTITY2Div extends COBJECT2Div{
    from
        co: openEHRMM!CDVQUANTITY,
            guiComp: archMM!EHRView,
                archetype: openEHRMM!ARCHETYPE
    to
        divVar: richUbiMM!Div (
            id <- co.rmTypeName,
            blockLevelComponents <- thisModule.
                createSpan(co.rmTypeName,co.rmTypeName),
            inlineComponents  <- spanVar
        ),
        spanVar : richUbiMM!Span(
            text <-   thisModule.
                getOntName(co.refImmediateComposite().
                    refImmediateComposite().
                        nodeId,archetype),
            inlineComponents <- phraseVar_value
        )
        ,
        phraseVar_value : richUbiMM!PhraseElement(
            text <- 'N/A', -- valor a ser resgatado em BD
            id <- co.refImmediateComposite().
                refImmediateComposite().nodeId
        )
}

lazy rule CDVQUANTITY2Fieldset extends COBJECT2Div{
```

```
    from
        co: openEHRMM!CDVQUANTITY,
            guiComp: archMM!EHRForm,
                archetype: openEHRMM!ARCHETYPE
    to
        divVar : richUbiMM!Div(
            id <- co.oclType().name+'---'+ co.rmTypeName,
            blockLevelComponents <- thisModule.
                createSpan(co.rmTypeName,co.rmTypeName),
            fieldsets <- fieldsetVar
        ),
        fieldsetVar : richUbiMM!Fieldset(
            legend <- thisModule.
                getOntName(co.refImmediateComposite().
                    refImmediateComposite().
                        nodeId,archetype),
            controls <- textFieldVar,
            controls <- selectVar
        ),
        textFieldVar : richUbiMM!TextField(
             name <- thisModule.
                getOntName(co.refImmediateComposite().
                    refImmediateComposite().nodeId,archetype)
        ),
        spanVar : richUbiMM!TextField(
             readonly <- true
        ),
        selectVar : richUbiMM!Select(
            items <- co.list -> collect(e | thisModule.
                CQUANTITYITEM2SelectItem(e))
        )
}

lazy rule CQUANTITYITEM2SelectItem{
    from
        cquantityitem : openEHRMM!CQUANTITYITEM
    to
        selectVar : richUbiMM!SelectItem(
             value <- cquantityitem.units,
             itemLabel <- cquantityitem.units
        )
}

-- Regras auxiliares imperativas --

rule createSpan(id: String, label: String){
    using {
        labelGroup : String = 'LabelGroup';
    }
    to divLabelGroup : richUbiMM!Div(
            id <- id + labelGroup,
```

```
            inlineComponents <- spanVar,
            class <- id + labelGroup,
            lineBreaks <- richUbiMM!LineBreak.newInstance(),
            lineBreaks <- richUbiMM!LineBreak.newInstance()
        ),
        spanVar : richUbiMM!Span (
            text <- label,
            id <- id + 'Span',
            class <- id + 'Span'
        )
    do{
        if (thisModule.checkLabel(id)){
            divLabelGroup;
        } else{
            OclUndefined;
        }
    }
}

rule CreateSelectItem(item : String, archetype: openEHRMM!
   ARCHETYPE){
    to
        selectVar : richUbiMM!SelectItem(
             value <- thisModule.getOntName(item,archetype),
             itemLabel <- thisModule.getOntName(item,archetype)
        )
    do{
        selectVar;
    }
}
```

FUNÇÕES AUXILIARES ATL (*HELPERS*)

Para oferecer suporte ao terceiro grupo de regras de transformação (modelos de dados clínicos) apresentado anteriormente, foram especificadas neste trabalho funções auxiliares para a geração de GUI associada a elementos visuais, como "rótulos", que referenciem a terminologia associada ao conceito clínico de interesse. As funções ATL apresentadas no código abaixo permitem capturar essa terminologia por meio do código associado ao conceito clínico no *open*EHR.

```
-- Este helper busca um determinado arquétipo.
helper context archMM!PropertyEHR def:
        getArchetype(id : String): openEHRMM!ARCHETYPE =
            openEHRMM!ARCHETYPE.allInstances()->
                select(e | e.archetypeId.value = id)->first();
```

```
-- Este helper resgata a terminologia referente ao código de um
   elemento de um arquétipo.
helper def: getOntName
    (id : String,obj : openEHRMM!ARCHETYPE): String =
        if
            id.startsWith('at')
        then
            obj.ontology.termDefinitions->
                select(x|x.language = 'en')->
                    first().items->select(y|y.code= id)->
                        first().items->
                            select(z|z.id='text')->first().
                                value
        else
            ''

    endif;

-- Este helper permite facilitar futuros processos de
   transformação de M2C.
helper def: checkLabel(s: String) : Boolean =
    if
        s='EVENT_data' or
            s='EVENT_state' or
                s='OBSERVATION_protocol' or
                    s='ARCHETYPE' or
                        s='EVALUATION_data' or
                            s='EVALUATION_protocol'

    then true
    else false
    endif;
```

REFERÊNCIAS

ATALAG, K.; YANG, H. Y. **From openEHR Domain Models to Advanced User Interfaces: A Case Study in Endoscopy**. 2010. Health Informatics New Zealand Conference.

BAUDRY, J. **ACME 1.2 ZooFederation**. 2006. Disponível em: www.emn.fr/z-info/atlanmod/index.php/ZooFederation#ACME_1.2 (Acesso: novembro de 2012).

BEALE, T. **Archetypes: Constraint-based domain models for future-proof information systems**. 2002. *openEHR* Standard document.

BEELER, G. W. Hl7 version 3: An object-oriented methodology for collaborative standards development. **International Journal of Medical Informatics**, v. 48, p. 151–161, 1998.

BLOBEL, B.; PHAROW, P. **Analysis and evaluation of EHR approaches.** 2009. Methods Inf Med 48(2), 162-9.

BRAGA, R. M.; CAVALINI, L. T.; CIRILO, C. E.; COOK, T. W.; CORREIA, B. S. P. M.; FREIRE, S. M.; GOMES, A. T. A.; MOREIRA, V. M.; MENEZES, A.; MORAES, J. L. C.; PRADO, A. F.; SOUZA, W. L.; TEIXEIRA, I. M.; ZIVIANI, A. Scientific computing applied to medicine and healthcare. In: _____, 2012. (Current State and Future Trends at the INCT-MACC the Brazilian National Institute of Science and Technology in Medicine Assisted by Scientific Computing, v. 1), cap. Model-Driven Development of Healthcare Applications, p. 315–354.

CANTRILL, S. V. Computers in patient care: The promise and the challenge. **Commun. ACM**, v. 53, p. 42–47, 2010.

CAVALINI, L. T.; COOK, T. W. Health informatics: The relevance of open source and multilevel modeling. In: HISSAM, S. A.; RUSSO, B.; NETO, M. G. de M.; KON, F. (Ed.). **OSS**, 2011. (IFIP Advances in Information and Communication Technology, v. 365), p. 338–347. ISBN 978-3-642-24417-9. Disponível em: <http://dblp.uni-trier.de/db/conf/oss/oss2011.html#CavaliniC11>.

CAVALINI, L. T.; GOMES, A. T. A.; ZIVIANI, A.; COOK, T. W. **Migração de um Sistema de Emergência Pré-Hospitalar de Cardiologia para a Modelagem Multinível**. 2012. Workshop de Informática Médica (WIM), 2012, Curitiba, PR - Brasill. Anais do XII Workshop de Informática Médica. Porto Alegre, RS - Brasil: SBC, 2012.

CHEN, H.; ZENG, D.; YAN, P.; YAN, P. **Infectious Disease Informatics: Syndromic Surveillance for Public Health and Biodefense**, 2010. (Integrated series in information systems). ISBN 9781441912787. Disponível em: <http://books.google.com.br/books?id=5BdCfSxtNJMC>.

CHRISTOPH, Z. Communication and interoperability for serial comparison in continuous health care - the new challenges. In: **Wearable eHealth Systems for Personalised Health Management: State of the Art and Future Challenges**, 2004. p. 172–180+. Disponível em: <http://iospress.metapress.com/content/qqjcjcpabem5u5nh>.

CIRILO, C. E.; PRADO, A. F.; SOUZA, W. L. D.; ZAINA, L. A. M. Model driven richubi: a model driven process for building rich interfaces of context-sensitive ubiquitous applications. **Proceedings of the 28th**, ACM, p. 207–214, 2010. Disponível em: <http://portal.acm.org/citation.cfm?id=1878485>.

CKM. **Clinical Knowledge Manager (CKM)**. 2012. Produzido por: Ocean Informatics. Disponível em: http://openehr.org/knowledge/ (Acessado em abril de 2012.).

CLEMENTS, P.; NORTHROP, L. **Software Product Lines: Practices and Patterns**, 2002. (SEI Series in Software Engineering). ISBN 9780201703320. Disponível em: <http://books.google.com.br/books?id=tHGFQgAACAAJ>.

CORREA, B. S. P. M.; GONçALVES, B.; TEIXEIRA, I. M.; GOMES, A. T. A.; ZIVIANI, A. Atoms: a ubiquitous teleconsultation system for supporting ami patients with prehospital thrombolysis. **Int. J. Telemedicine Appl.**, Hindawi Publishing Corp., New York, NY, United States, v. 2011, p. 2:1–2:12, jan. 2011. ISSN 1687-6415. Disponível em: <http://dx.doi.org/10.1155/2011/560209>.

COSTA, S. L. da. **Uma Abordagem Baseada em Modelos para Construção Automática de Interfaces de Usuário para Sistemas de Informação**. 2011. Dissertação de Mestrado. Goiânia, UFG.

CZARNECKI, K.; ANTKIEWICZ, M.; KIM, C. H. P.; LAU, S.; PIETROSZEK, K. Model-driven software product lines. In: **Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**, 2005. (OOPSLA '05), p. 126–127. ISBN 1-59593-193-7. Disponível em: <http://doi.acm.org/10.1145/1094855.1094896>.

CZARNECKI, K.; HELSEN, S. Feature-based survey of model transformation approaches. **IBM Systems Journal**, v. 45, n. 3, p. 621–646, 2006. Disponível em: <http://dx.doi.org/10.1147/sj.453.0621>.

Eclipse Foundation. **XML Schema to Ecore Mapping**. 2004. Disponível em: http://www.eclipse.org/modeling/emf/docs/overviews/XMLSchemaToEcoreMapping.pdf (Acesso: dezembro de 2012).

Eclipse Foundation. **EMF: Eclipse Modeling Framework**. 2012. Disponível em: http://www.eclipse.org/modeling/emf/ (Acesso: novembro 2012).

EICHELBERG, M.; ADEN, T.; RIESMEIER, J.; DOGAC, A.; LALECI, G. B. A survey and analysis of electronic healthcare record standards. **ACM Comput. Surv.**, ACM, New York, NY, USA, v. 37, n. 4, p. 277–315, dez. 2005. ISSN 0360-0300. Disponível em: <http://doi.acm.org/10.1145/1118890.1118891>.

FABRO, M. D. D.; BEZIVIN, J.; JOUAULT, F.; BRETON, E.; GUELTAS, G. Amw: a generic model weaver. In: **Proceedings of the 1ère Journée sur l'Ingénierie Dirigée par les Modèles (IDM05)**, 2005. Disponível em: <http://www.sciences.univ-nantes.fr/lina/atl/www/papers/IDM_2005_weaver.pdf>.

GAMMA, E.; HELM, R.; JOHNSON, R.; VLISSIDES, J. **Design Patterns**, 1995. Hardcover. ISBN 0201633612. Disponível em: <http://www.amazon.co.uk/exec/obidos/ASIN/0201633612/citeulike-21>.

GARLAN, D.; MONROE, R.; WILE, D. Acme: An architecture description interchange language. In: **in Proceedings of CASCON'97**, 1997. p. 169–183.

GOMES, A. T. A.; ZIVIANI, A.; CORREA, B. S. P. M.; TEIXEIRA, I. M.; MOREIRA, V. M. Splice: a software product line for healthcare. In: **Proceedings of the 2nd ACM SIGHIT International Health Informatics Sym-**

posium, 2012. (IHI '12), p. 721–726. ISBN 978-1-4503-0781-9. Disponível em: <http://doi.acm.org/10.1145/2110363.2110447>.

HAUX, R. Medical informatics: Past, present, future. **I. J. Medical Informatics**, v. 79, n. 9, p. 599–610, 2010. Disponível em: <http://dblp.uni-trier.de/db/journals/ijmi/ijmi79.html#Haux10>.

HL7 CDA. **The Health Level 7 Version 3 Standard: Clinical Data Architecture. Release 2.0. ANSI Standard.** 2005.

HOOT, N.; ARONSKY, D. Systematic review of emergency department crowding: Causes, effects, and solutions. **Annals of Emergency Medicine**, v. 52, n. 2, p. 126–136.e1, ago. 2008. ISSN 01960644. Disponível em: <http://dx.doi.org/10.1016/j.annemergmed.2008.03.014>.

HUDSON, D. L.; COHEN, M. E. Uncertainty and complexity in personal health records. **Conf Proc IEEE Eng Med Biol Soc**, v. 1, p. 6773–6, 2010. ISSN 1557-170X. Disponível em: <http://www.biomedsearch.com/nih/Uncertainty-complexity-in-personal-health/21095837.html>.

JANAMANCHI, B.; KATSAMAKAS, E.; RAGHUPATHI, W.; GAO, W. The state and profile of open source software projects in health and medical informatics. **I. J. Medical Informatics**, v. 78, n. 7, p. 457–472, 2009. Disponível em: <http://dblp.uni-trier.de/db/journals/ijmi/ijmi78.html#JanamanchiKRG09>.

JOB A. T. A. GOMES, e. A. Z. D. H. Telemedicine and e-health services, policies and applications: Advancements and developments. In: _____, 2011. cap. Health Systems for Syndromic and Epidemiological Surveillance., p. 245–262.

JOUAULT, F.; ALLILAIRE, F.; BÉZIVIN, J.; KURTEV, I. ATL: A model transformation tool. **Sci. Comput. Program.**, Elsevier North-Holland, Inc., Amsterdam, The Netherlands, The Netherlands, v. 72, n. 1-2, p. 31–39, jun. 2008. ISSN 0167-6423. Disponível em: <http://dx.doi.org/10.1016/j.scico.2007.08.002>.

KEELEY, E. C.; BOURA, J. A.; GRINES, C. L. Primary angioplasty versus intravenous thrombolytic therapy for acute myocardial infarction: a quantitative review of 23 rando-

mised trials. **The Lancet**, v. 361, n. 9351, p. 13 – 20, 2003. ISSN 0140-6736. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0140673603121137>.

LINDEN, H. V. D.; AUSTIN, T.; TALMON, J. Generic screen representations for future-proof systems, is it possible? there is more to a gui than meets the eye. **Computer Methods and Programs in Biomedicine**, Conseil sur cybernétique, Académie des Sciences de l'URSS, v. 95, n. 3, p. 213–226, 2009. Disponível em: <http://discovery.ucl.ac.uk/149737/>.

LOPEZ, D. M.; BLOBEL, B. A development framework for semantically interoperable health information systems. **I. J. Medical Informatics**, v. 78, n. 2, p. 83–103, 2009. Disponível em: <http://dblp.uni-trier.de/db/journals/ijmi/ijmi78.html#LopezB09>.

MARTIN, J.; ODELL, J. **Object-oriented methods: a foundation**, 1998. (The James Martin Books). ISBN 9780139055973. Disponível em: <http://books.google.com.br/books?id=JotQAAAAMAAJ>.

MCGINN, C.; GRENIER, S.; DUPLANTIE, J.; SHAW, N.; SICOTTE, C.; MATHIEU, L.; LEDUC, Y.; LEGARE, F.; GAGNON, M.-P. Comparison of user groups' perspectives of barriers and facilitators to implementing electronic health records: a systematic review. **BMC Medicine**, v. 9, n. 1, p. 46, 2011. ISSN 1741-7015. Disponível em: <http://www.biomedcentral.com/1741-7015/9/46>.

MENÁRGUEZ-TORTOSA, M.; MARTÍNEZ-COSTA, C.; FERNÁNDEZ-BREIS, J. T. A generative tool for building health applications driven by iso 13606 archetypes. **J Med Syst**, 2011. ISSN 0148-5598. Disponível em: <http://www.biomedsearch.com/nih/Generative-Tool-Building-Health-Applications/21968574.html>.

NARDON, F. B.; FRANçA, T.; NAVES, H. Construção de aplicações em saúde baseadas em arquétipos. **Acta Scientiarum Biological Sciences**, v. 28, n. 4, 2007.

NHS. **National Health Service Media Centre: Dismantling the nhs national programme for it**. 2011.

OMG. **Unified Modeling Language (UML), Infrastructure**. 2003. Disponível em: http://www.omg.org/spec/UML/2.3/Infrastructure/PDF/ (Acesso: dezembro de 2012).

OMG. **Meta Object Facility (MOF) Core Specification Version 2.0**. 2006. Disponível em: http://www.omg.org/cgi-bin/doc?formal/2006-01-01 (Acesso: abril de 2012).

OMG. **Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification**. 2008. Disponível em: http://www.omg.org/spec/QVT/1.0/PDF/ (Acesso: novembro 2012).

OMG. **Object Management Group (OMG)**. 2012. Disponível em: http://www.omg.org (Acesso: novembro de 2012).

RAGHUPATHI, W.; UMAR, A. Exploring a model-driven architecture (MDA) approach to health care information systems development. **Int J Med Inform**, v. 77, p. 305–14, 2008.

SASSER, S.; VARGHESE, M.; KELLERMANN, A.; LORMAND, J.-D. **Prehospital trauma care systems**, 2005.

SCHULER, T.; GARDE, S.; HEARD, S.; BEALE, T. Towards automatically generating graphical user interfaces from openehr archetypes. In: HASMAN, A.; HAUX, R.; LEI, J. van der; CLERCQ, E. D.; FRANCE, F. H. R. (Ed.). **MIE**, 2006. (Studies in Health Technology and Informatics, v. 124), p. 221–226. ISBN 978-1-58603-647-8. Disponível em: <http://dblp.uni-trier.de/db/conf/mie/mie2006.html#SchulerGHB06>.

SHALOM, E.; SHAHAR, Y.; TAIEB-MAIMON, M.; MARTINS, S.; VASZAR, L.; GOLDSTEIN, M.; GUTNIK, L.; LUNENFELD, E. Ability of expert physicians to structure clinical guidelines: reality versus perception. **J Eval Clin Pract**, v. 15, n. 6, p. 1043–53, 2009.

SMITH, A. M.; HARDY, P. J.; SANDLER, D. A.; COOKE, J. Paramedic decision making: prehospital thrombolysis and beyond. **Emerg Med J**, v. 28, n. 8, p. 700–2, 2011. ISSN 1472-0213. Disponível em: <http://www.biomedsearch.com/nih/Paramedic-decision-making-prehospital-thrombolysis/20682955.html>.

STAHL, T.; VÖLTER, M. **Model-Driven Software Development: Technology, Engineering, Management**, 2006. ISBN 978-0-470-02570-3.

TUOMAINEN, M. P.; MYKKÄNEN, J. A.; LUOSTARINEN, H.; PÖYHÖLÄ, A.; PAAKKANEN, E. Model-centric approaches for the development of health information systems. In: KUHN, K. A.; WARREN, J. R.; LEONG, T.-Y. (Ed.). **MedInfo**, 2007. (Studies in Health Technology and Informatics, v. 129), p. 28–32. ISBN 978-1-58603-774-1. Disponível em: <http://dblp.uni-trier.de/db/conf/medinfo/medinfo2007.html#TuomainenMLPP07>.

UK Biobank. **United Kingdom Biobank: a large-scale prospective epidemiological resource**. 2012. Disponível em: http://www.p3gobservatory.org/catalogue.htm?questionnaireId=9 (Acesso: dezembro de 2012).

VEST, J. R. More than just a question of technology: factors related to hospitals? adoption and implementation of health information exchange. **International Journal of Medical Informatics**, Elsevier Ireland Ltd, v. 79, n. 12, p. 797–806, 2010. Disponível em: <http://www.ncbi.nlm.nih.gov/pubmed/20889370>.

WHO. **World Health Organization**. 2012. Disponível em: http://www.who.int/en/ (Acesso: dezembro de 2012).

World Wide Web Consortium (W3C). **Cascading Style Sheets (CSS)**. 2012. Disponível em: http://www.w3.org/Style/CSS/ (Acesso: dezembro de 2012).