Estudo da Viabilidade de Utilização o *Framework* GREN para Instanciar Aplicações no Domínio de Clínicas de Reabilitação

Anderson Pazin[1]
Faculdades Salesiana de Lins, Centro de Tecnologia de Informação.
Lins - SP, Brasil, CEP: 16400-505, 55(0xx14) 3522 - 4733
anderson@salesianolins.br

Ricardo Argenton Ramos
Rosângela Ap. D. Penteado
[1]UFSCar - Universidade Federal de São Carlos, Departamento de Computação
São Carlos - SP, Brasil, CEP:13565-905, 55(0xx 16) 260-8233
{ rar, rosangel}@dc.ufscar.br

Abstract

The domain of rehabilitation clinics management can be considered as a sub domain of bussiness resource management. The GREN framework instanced applications of the domain bussiness resource management using the pattern language GRN. However, such framework does not deal with specific functionalities of the domain of rehabilitation clinics management, as for example the accompaniment of the treatment at patient. A pattern language for this domain, called SiGCli (Sistemas para Gerenciamento de Clínicas in protuguese) was elaborated on the basis of the GRN. Some patterns of the GRN had been adapted to the specific necessities of the SiGCli domain. This paper presents the use of GREN framework to instanced applications of the domain of rehabilitation clinics management, being commented the advantages and disadvantages of this use.

Key Words: pattern language, management of rehabilitation clinics, bussiness resource, reuse.

Resumo

*O domínio de gerenciamento de clinicas de reabilitação, pode ser considerado como um subdomínio de gestão de recursos de negócios. O framework GREN instancia aplicações do domínio de gestão de recursos de negócios utilizando a linguagem de padrões GRN. Entretanto, tal framework não trata de funcionalidades específicas do domínio de gerenciamento de clinicas de reabilitação, como por exemplo o acompanhamento do tratamento de um paciente. Uma linguagem de padrões para esse domínio, denominada SiGCli (**Si**stemas para **G**erenciamento de **Cli**nicas de reabilitação) foi elaborada com base na GRN. Alguns padrões da GRN foram adaptados às necessidades específicas do domínio SiGCli. Este trabalho apresenta a utilização do framework GREN para instanciar aplicações do domínio de clinicas de reabilitação, comentando as vantagens e desvantagens dessa utilização.*

Palavras-Chave: linguagem de padrões, gerenciamento de clínicas de reabilitação, gestão de negócios, reuso.

.

1. Introdução

O reuso de software é uma atividade comum em empresas de desenvolvimento, devido à necessidade de melhorar a produtividade, a manutenibilidade e a qualidade tanto do software quanto do processo de desenvolvimento. Uma forma de melhorar o reuso é utilizar padrões de software, por possibilitarem o reuso em vários níveis de abstração (código, projeto, análise, arquitetura e processo de desenvolvimento) [2].

Padrões de software descrevem soluções de sucesso para os problemas de desenvolvimento de software que geralmente ocorrem em determinados contextos, com a finalidade de auxiliar os desenvolvedores menos experientes com os conhecimentos dos mais experientes, para que possam agir como especialistas [3].

Uma linguagem de padrões é uma coleção de padrões organizados que se apóiam entre si para transformar requisitos e restrições numa arquitetura [4]. Os padrões que a constituem devem abranger todos os aspectos importantes de um determinado domínio e pelo menos um padrão deve estar disponível para cada aspecto da construção e implementação de um sistema de software. A linguagem de padrões auxilia na subdivisão de problemas gerais com soluções complexas em problemas menores e relacionados, de forma a facilitar a solução. Uma característica importante para as linguagens de padrões é que cada padrão pode ser usado de forma isolada ou com um certo número de padrões da linguagem, com isso um único padrão é considerado útil mesmo que a linguagem não seja aplicada em sua totalidade.

Um *framework* é uma infra-estrutura genérica, baseada em um domínio, que pode ser adaptada para solucionar problemas específicos desse domínio, servindo como um modelo para a construção de aplicações através da especificação das classes e das colaborações entre elas. Um *framework* reusa: a) análise, pois descreve os tipos de objetos importantes e como um problema maior pode ser dividido em problemas menores; b) projeto, pois contém algoritmos abstratos e descreve a interface que deve ser implementada, e as restrições a serem satisfeitas pela implementação; e c) código, pois se torna mais fácil desenvolver uma biblioteca de componentes compatíveis e porque a implementação desses pode herdar grande parte de seu código das super-classes abstratas. Apesar de todos esses tipos de reuso serem importantes, os reusos de análise e de projeto são os que mais apresentam vantagens em longo prazo [5,6].

Esse artigo apresenta os resultados obtidos ao se aplicar uma linguagem de padrões específica (SiGCli), definida para um subdomínio de gestão de recursos de negócios, utilizando o framework GREN,desenvolvido com base na linguagem de padrões GRN, para instanciar aplicações desse subdomínio. A seção 2 apresenta a linguagem de padrões GRN e o *framework* GREN, a seção 3 trata do processo de elaboração a linguagem de padrões SiGCli [7] com base em outra linguagem de padrões. A aplicação dos padrões da SiGCli com *framework* GREN é mostrada na seção 4 e as considerações finais sobre o estudo realizado são apresentadas na seção 5.

2. Assuntos Relacionados

2.1. A Linguagem de Padrões GRN.

A Linguagem de Padrões para Gestão de Recursos de Negócios (GRN), [8,9,10] é composta por quinze padrões de análise, sendo alguns deles aplicações ou extensões de padrões existentes na literatura. Essa linguagem auxilia desenvolvedores menos experientes no desenvolvimento de aplicações que tratam de gestão de recursos de negócios, ou seja, nas quais existe a necessidade de registrar transações de aluguel, comércio ou manutenção de recursos. A Figura 1, extraída de [10] exemplifica a GRN, mostrando os relacionamentos e dependências entre os padrões. A ordem de aplicação desses padrões é definida, bem como as dependências existentes entre eles, representadas pela descrição do elemento "Próximos Padrões". Na Figura 1, as linhas mais espessas indicam os principais padrões da linguagem: LOCAR O RECURSO, COMERCIALIZAR O RECURSO e MANTER O RECURSO.

Os padrões estão agrupados de acordo com seu propósito. No primeiro grupo estão três padrões: IDENTIFICAR O RECURSO, QUANTIFICAR O RECURSO e ARMAZENAR O RECURSO, que tratam da identificação e possível qualificação, quantificação e armazenagem dos recursos gerenciados pelo negócio. No segundo grupo estão os padrões relacionados à manipulação dos recursos de negócio pelo sistema. Existem sete padrões nesse grupo: LOCAR O RECURSO, RESERVAR O RECURSO, COMERCIALIZAR O RECURSO, COTAR O RECURSO, CONFERIR A ENTREGA DO RECURSO, MANTER O RECURSO e COTAR A MANUTENÇÃO. O terceiro grupo possui cinco padrões que cuidam de detalhes das transações efetuadas com o recurso. Os três primeiros, ITEMIZAR TRANSAÇÃO DO RECURSO, PAGAR PELA TRANSAÇÃO DO RECURSO e IDENTIFICAR O EXECUTOR DA TRANSAÇÃO, são aplicáveis a quaisquer das transações do grupo 2. Os dois últimos, IDENTIFICAR AS TAREFAS DA MANUTENÇÃO e IDENTIFICAR AS PEÇAS DA MANUTENÇÃO, são aplicáveis às transações contidas nos padrões MANTER O RECURSO e COTAR A MANUTENÇÃO.

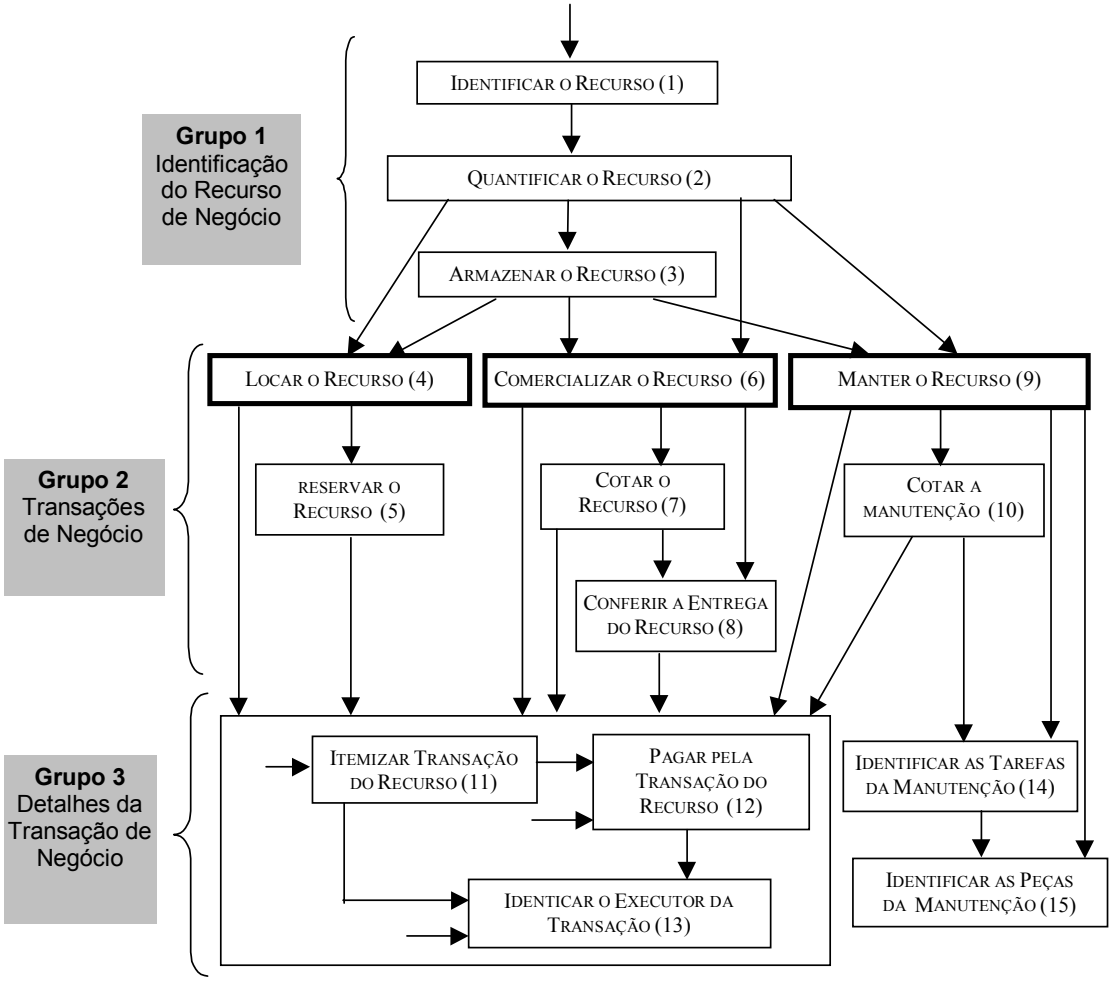

Figura 1. Relacionamento entre os padrões da linguagem GRN [10]

2.2. O *Framework* GREN.

Braga e Masiero [1] propõem um processo de construção de *frameworks* com base em uma linguagem de padrões. Dessa forma, a GRN foi utilizada para elaborar o *framework* GREN, desenvolvido em linguagem *Smalltalk*, para instanciação de aplicações no domínio de sistemas de gestão de recursos de negócios. A Figura 2 mostra a arquitetura do *Framework* GREN.

A camada *GREN-Persistência* possui classes para cuidar da conexão com a base de dados, gerenciamento dos identificadores dos objetos e persistência dos objetos. A camada *GREN-Negócios* comunica-se com a camada *GREN-Persistência* sempre que for necessário armazenar permanentemente um objeto. Dentro da camada *GREN-Negócios* existem diversas classes derivadas diretamente dos padrões de análise que compõem a GRN, ou seja, as classes e os relacionamentos contidos em cada padrão possuem a implementação correspondente nessa camada. A camada *GREN-Interface Gráfica com o Usuário* contém as classes responsáveis pela entrada e saída de dados, como formulários de interface, janelas e menus, que permitem a interação do usuário final com o sistema. Essa camada comunica-se com a *GREN-Negócios* para obtenção de objetos a serem mostrados na interface com o usuário. A camada *GREN-Wizard*, encontra-se acima da camada de interface gráfica com o usuário, pois utiliza todas as demais camadas para realizar a instanciação de aplicações de forma gráfica com base na GRN.

As aplicações específicas podem ser instanciadas a partir da camada *GREN-Interface Gráfica com o Usuário* usando, por meio de herança ou referência a objetos, classes de todas as camadas do GREN. Quando não existem classes a serem reutilizadas da camada *GREN-Interface Gráfica com o Usuário*, a instanciação de uma aplicação é realizada a partir da *GREN-Negócios*, sendo a interface implementada separadamente. Outra opção é a partir do *GREN-Wizard*, que se encarrega de fazer a comunicação com as demais camadas do GREN.

As instanciações das aplicações no GREN são guiadas pela aplicação da Linguagem de Padrões GRN, que gera diagramas de classes da aplicação desejada. Com base nesses diagramas utilizam-se classes pré-programadas do *framework* GREN para criar o código da nova aplicação.

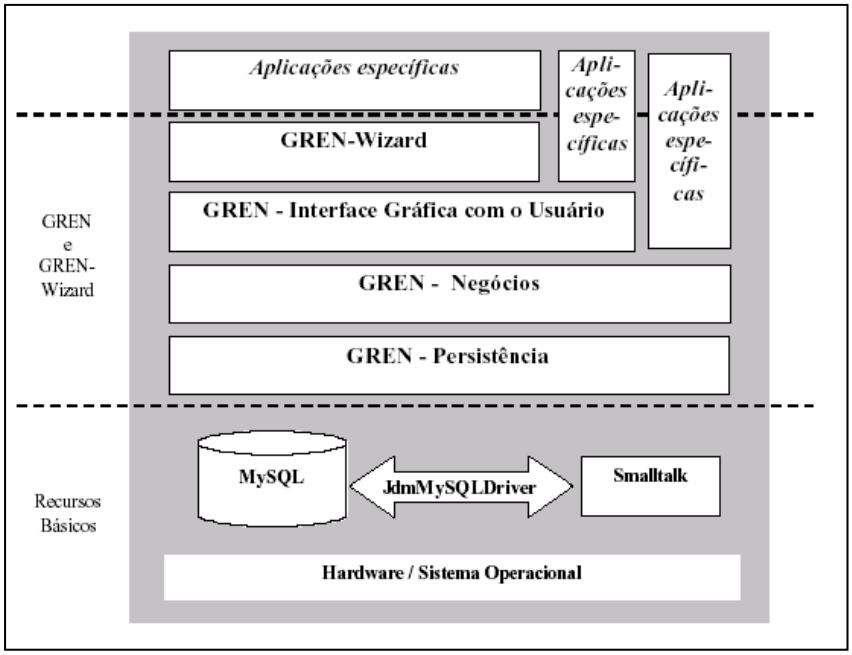

Figura 2. Arquitetura do Framework GREN [10]

3. A Linguagem de Padrões SiGCli

A linguagem de padrões para **Si**stemas de **G**erenciamento de **Cl**ínicas de Reabilitação, denominada SiGCli, foi elaborada com base no mesmo processo usado para a elaboração da GRN [10], Figura 3.

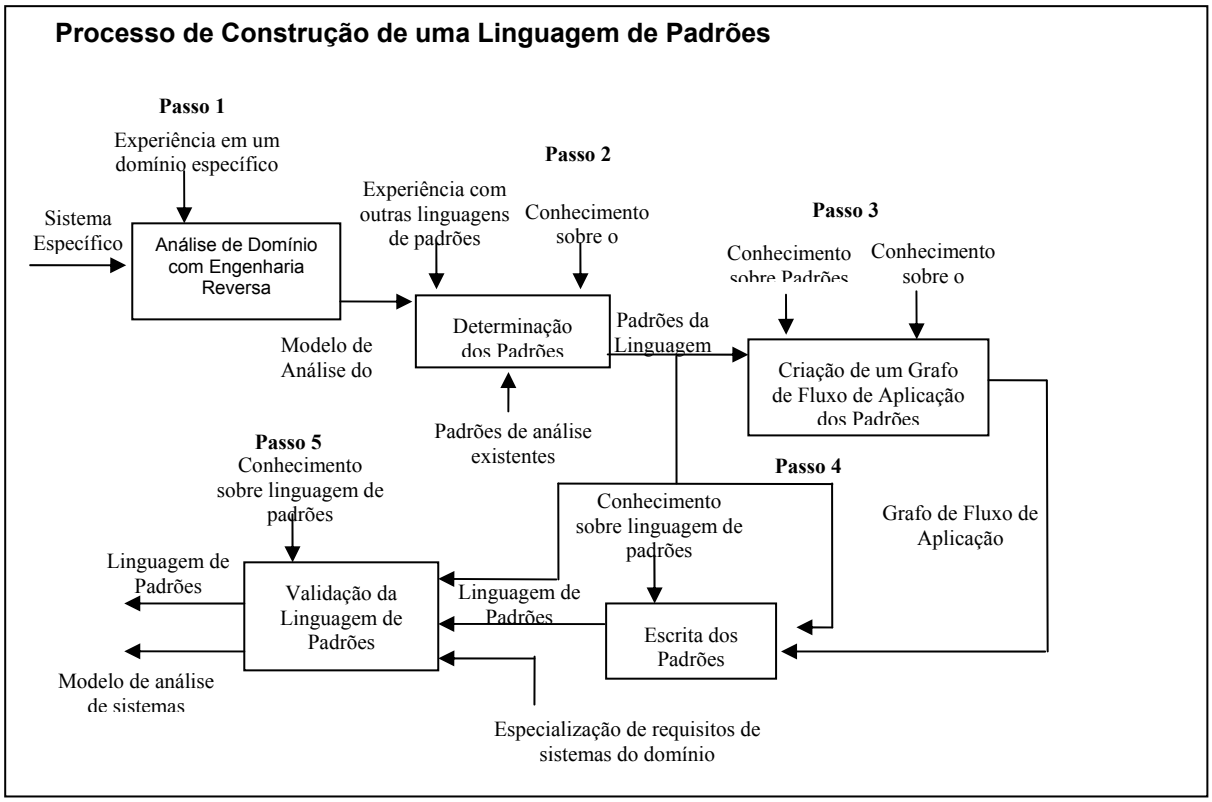

Figura 3. Processo de construção de uma linguagem de padrões com base em análise de domínio [10]

O passo 1, *Análise de Domínio com Engenharia Reversa*, foi realizado com base na engenharia reversa de sistemas legados. Roberts e Johnson [13] sugerem que o desenvolvimento de um *framework* deve acontecer após o desenvolvimento de três aplicações concretas que irão fornecer conhecimento suficiente para permitir de forma

gradativa a generalização dessas aplicações. Embora o objetivo deste passo não seja a elaboração de um *framework*, esses conceitos podem ser aplicados para a análise de domínio.Os sistemas escolhidos são sistemas reais que atualmente são utilizados pelas Clínicas de Reabilitação Física Dom Bosco, das Faculdades Salesianas de Lins - SP. O processo usado para engenharia reversa desses sistemas é específico, pois são desenvolvidos em ambiente ZIM [11]. Dois apoios computacionais, `Gera Java` e `Gera SQL`, são utilizados para facilitar a recuperação desses sistemas, a partir das informações existentes no dicionário de dados das aplicações, gerando arquivos com finalidades específicas. `Gera Java` lê as definições do dicionário de dados e gera um arquivo, chamado `Classes.java`, considerando cada tabela ZIM, do sistema legado, como uma classe. `Gera SQL` lê as definições do dicionário de dados e gera três arquivos (`fldsdocs, relsdoc, Tabelas`) que facilitam a identificação das tabelas para o novo sistema. O processo de engenharia reversa pode ser apoiado por uma ferramenta CASE, como *Rational Rose*[12], que disponibilize o recurso de conversão de classes Java para um diagrama de classes. Maiores detalhes sobre o processo de engenharia reversa podem ser encontrados em [7].

O Modelo de Domínio é elaborado buscando-se a similaridade existente entre as classes de cada aplicação. Assim, tem-se o modelo do domínio exibido na Figura 4, com letras inseridas em cada classe para representar as similaridades observadas entre os três sistemas: S – classes Similares existentes nos três sistemas; PS – classes Parcialmente Similares existentes em dois dos três sistemas e E – classes Exclusivas que existem somente em um dos sistemas.

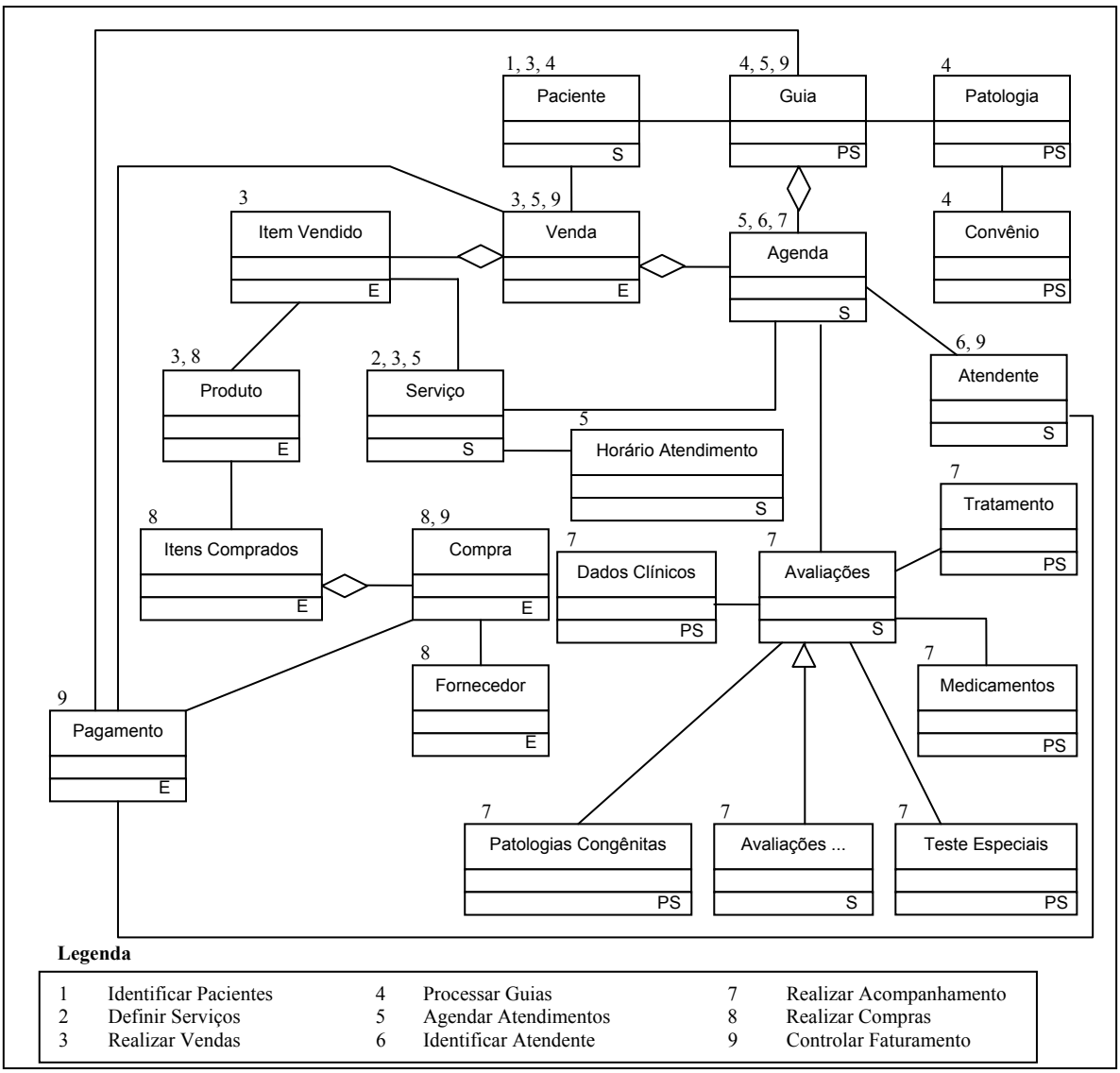

Figura 4. Modelo de classes do domínio para sistemas de gerenciamento de clínicas de reabilitação(SiGCLI).

O passo 2, *Determinação dos Padrões*, agrupa as classes do modelo do domínio, de acordo com suas funcionalidades, formando um conjunto de classes que determinará os futuros padrões da linguagem. Analisando-se o funcionamento dos sistemas escolhidos, verificou-se quais classes são necessárias para tratar de funcionalidades específicas agrupando e identificando-as por um nome, que define o padrão. Na Figura 4, toda classe possui uma numeração, no canto superior esquerdo, que identifica a qual padrão ela pertence. As classes pertencentes a mais de

um padrão, possuem números separados por vírgulas. A Legenda apresenta o nome dos padrões com seus respectivos números de identificação. Por exemplo, *Realizar Vendas* (3), é formado pelas classes *Paciente*, *Produto, Serviço, Venda, Item Vendido.*

O passo 3, *Criação de um Grafo de Fluxo de Aplicação dos Padrões*, prevê a definição da ordem de aplicação e interação entre os padrões da SiGCli, Figura 5. Uma das formas para a elaboração do grafo é iniciar pelos padrões que representam as funcionalidades mais básicas do domínio. Três grupos são formados: *Informações Básicas*, *Atendimentos* e *Controle de Transações Financeiras*, com os padrões extraídos do diagrama de classes apresentado na Figura 4 e especificados na Legenda.

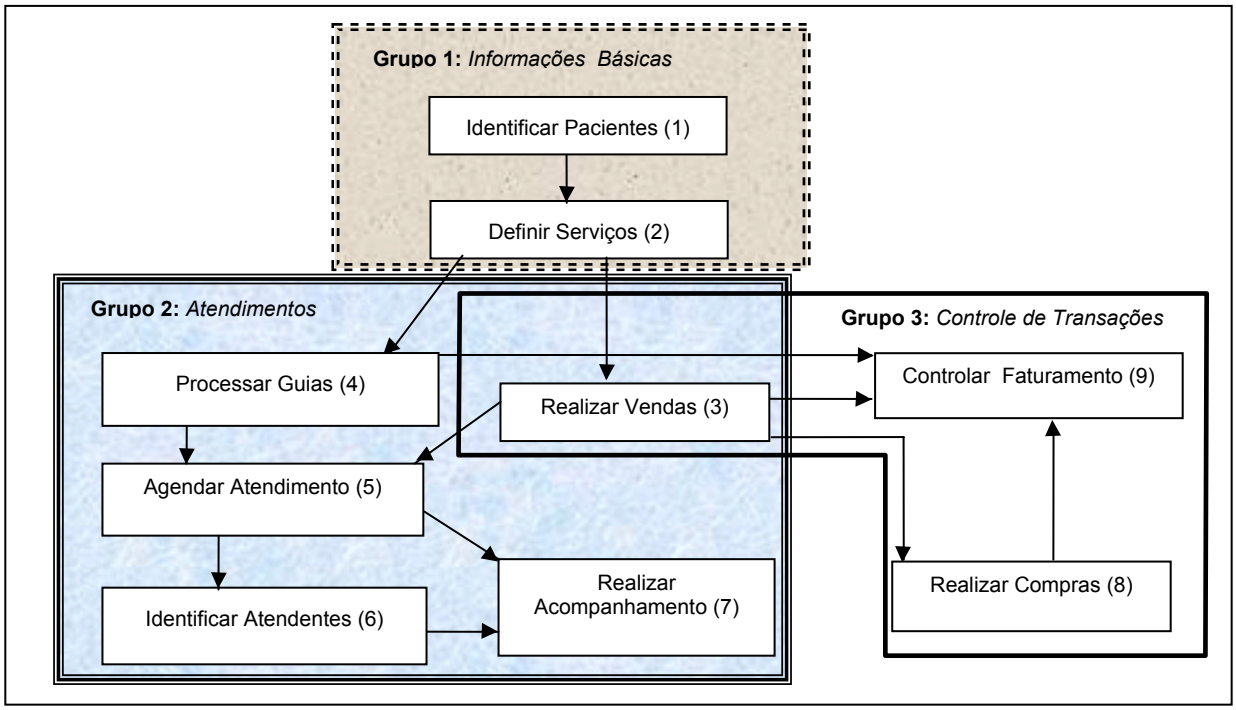

Figura 5. Grafo do fluxo de aplicação dos padrões da SiGCLI

O passo 4, *Escrita dos Padrões,* é o passo mais importante, uma vez que o padrão deve ser bem elaborado para que não haja dúvidas quanto à sua interpretação pelos analistas que o utilizarão. Existem alguns formatos propostos na literatura, sendo que o adotado foi o seguinte: "nome", "contexto", "problemas", "estrutura", "participantes", "padrões relacionados". O passo 5, *Validação da Linguagem de Padrões,* objetiva aplicar os padrões a diferentes sistemas do domínio estudado, comprovando sua eficiência.

Durante a realização dos passos 4 e 5 observou-se que o domínio da SiGCli apresenta pontos comuns com o domínio da GRN, assim, procedeu-se um mapeamento entre os padrões das duas linguagens. O domínio da SiGCLi possui algumas funcionalidades específicas que diferem das aplicações do domínio da GRN. Dessa forma, algumas adaptações nos padrões da GRN são necessárias, caracterizando a SiGCli como um sub-domínio da GRN.

A Tabela 1 exibe o mapeamento entre os padrões da SiGCli e os da GRN quando aplicados a sistemas de clínicas de reabilitação. O padrão 7 da SiGCli não possui mapeamento direto na GRN. Sua funcionalidade é especifica para o domínio de clínicas de reabilitação devido à necessidade de acompanhamento de todas as sessões realizadas por um paciente durante o seu tratamento nas clinicas de reabilitação.

Tabela 1- Mapeamento entre os padrões da SiGCli com os da GRN com foco no tratamento do Paciente.

SiGCli		**GRN**	
Nº Padrão	Padrão	Nº Padrão	Padrão
1	Identificar Pacientes	1	Identificar o Recurso
		2	Quantificar o Recurso
2	Definir Serviços	1	Identificar o Recurso
		2	Quantificar o Recurso
3	Realizar Vendas	1	Identificar o Recurso
		2	Quantificar o Recurso
		6	Comercializar o Recurso
		11	Itemizar Transação do Recurso
4	Processar Guias	9	Manter o Recurso

5	Agendar Atendimentos	14	Identificar Tarefas de Manutenção
6	Identificar Atendentes	13	Identificar o Executor da Transação
7	Realizar Acompanhamento		*Não possui mapeamento*
8	Realizar Compras	6	Comercializar o Recurso
		11	Itemizar Transação do Recurso
9	Controlar Faturamento	12	Pagar pela Transação do Recurso

O mapeamento ilustrado na Tabela 1 foi elaborado com o foco no paciente que procura uma clínica para se reabilitar de uma lesão. Nesse caso, tem-se que um recurso, o paciente, deve sofrer a manutenção, tratamento, realizada pela clínica. A GRN possibilita um outro foco com base nos serviços oferecidos pela clínica. Nessa situação tem-se que um recurso, o serviço oferecido pela clínica, é alugado para um determinado paciente para realizar o tratamento. Esse mapeamento é o ilustrado na Tabela 2. Como se pode observar, analisando as duas tabelas, apenas os padrões 3 e 4 da SiGCli sofreram alterações com relação à aplicação padrões da GRN. Usando as informações dessas tabelas a instanciação das aplicações com o *framework* GREN é facilitada, uma vez que os padrões já foram mapeados cabendo ao engenheiro de software somente aplicá-los conforme a definição do GREN-Wizard.

Tabela 2- Mapeamento entre os padrões da SiGCli com os da GRN com foco no aluguel de serviços da clínica.

SiGCli		**GRN**	
Nº Padrão	Padrão	Nº Padrão	Padrão
1	Identificar Pacientes	1	Identificar o Recurso
		2	Quantificar o Recurso
2	Definir Serviços	1	Identificar o Recurso
		2	Quantificar o Recurso
3	Realizar Vendas	1	Identificar o Recurso
		2	Quantificar o Recurso
		6	Comercializar o Recurso
		11	Itemizar Transação do Recurso
4	Processar Guias	5	Reservar o Recurso
5	Agendar Atendimentos	4	Locar o Recurso
6	Identificar Atendentes	13	Identificar o Executor da Transação
7	Realizar Acompanhamento		*Não possui mapeamento*
8	Realizar Compras	6	Comercializar o Recurso
		11	Itemizar Transação do Recurso
9	Controlar Faturamento	12	Pagar pela Transação do Recurso

4. Estudo de Caso: Utilização do *Framework* GREN para Instanciar Aplicações no Domínio SiGCli.

Esta seção apresenta um estudo de caso exemplificando o uso do *framework* GREN para instanciação de uma Clínica de Fisioterapia. Essa clínica controla todos os atendimentos realizados em seus pacientes e o seu objetivo principal é armazenar os acompanhamentos e avaliações realizadas pelos atendentes. Esses atendentes são alunos do curso de Fisioterapia e devem ser avaliados na disciplina de estágio supervisionado, existente em sua grade curricular. Esses atendentes não são remunerados e o controle do faturamento da clínica também não é de interesse nesse sistema. O modelo de classes do sistema é apresentado na Figura 6, sendo resultado da aplicação dos padrões (1) *Identificar Pacientes*, (2) *Definir Serviços*, (4) *Processar Guias*, (5) *Agendar Atendimentos*, (6) *Identificar Atendentes* e (7) *Realizar Acompanhamento* da linguagem de padrões SiGCli.

A instanciação da aplicação no *framework* GREN segue o grafo de fluxo de aplicação dos padrões da GRN que é diferente do da SiGCli. Logo, os padrões sem correspondência nessas duas linguagens devem ser instanciados manualmente pelo desenvolvedor, o que nem sempre é uma tarefa trivial.

A Tabela 3 define o mapeamento entre as classes disponíveis no *framework* GREN e as classes existentes na linguagem de padrões SiGCli. Somente o padrão 7 da SiGCli, *Realizar Acompanhamento*, não possui correspondência para a instanciação usando o *framework*. Cada classe no GREN tem um papel definido quando se pretende instanciar uma aplicação do domínio SiGCli. Dessa forma, para instanciar a aplicação apresentada pela

Figura 6, têm-se os padrões da SiGCli aplicados na seguinte ordem (1), (4), (6), (5), (2), (7) de acordo com o grafo de fluxo de aplicação dos padrões da GRN.

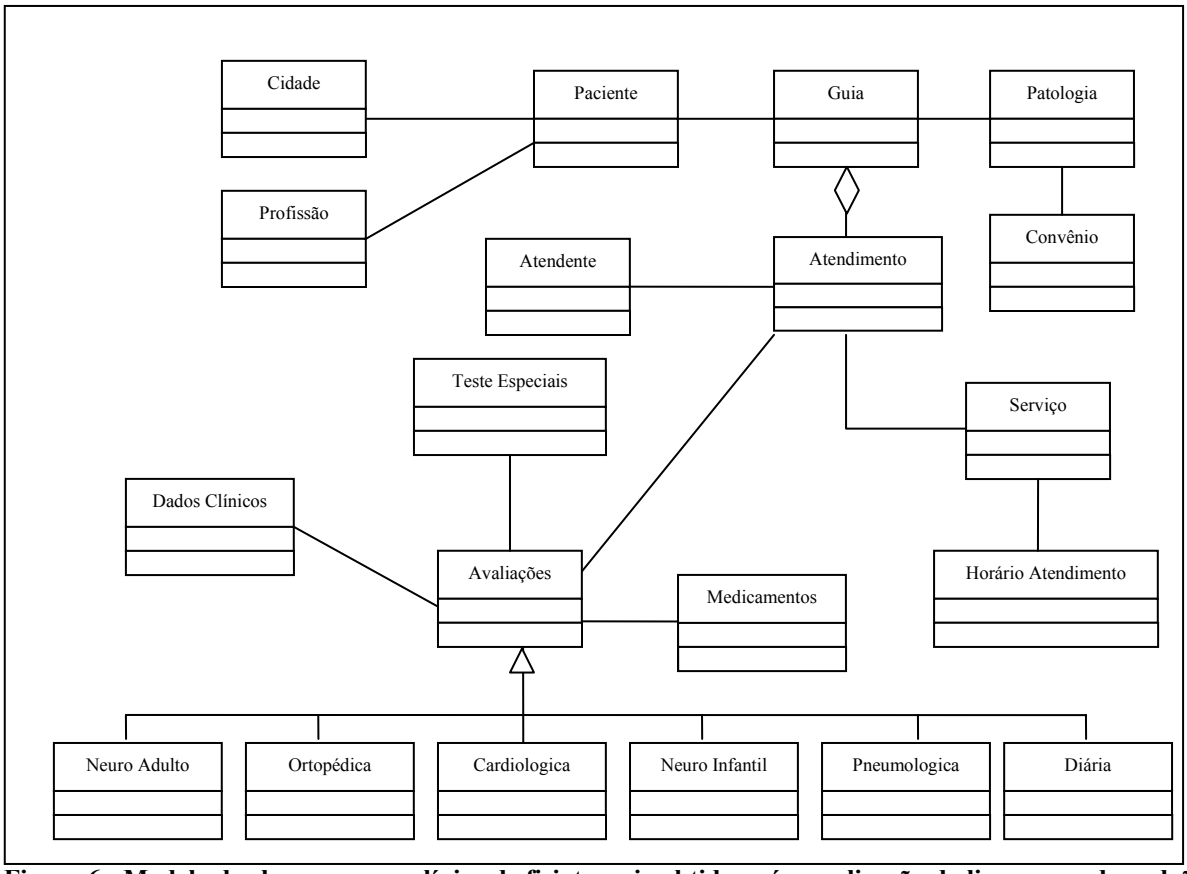

Figura 6 - Modelo de classes para a clínica de fisioterapia obtido após a aplicação da linguagem de padrões SiGCli.

Tabela 3 - Mapeamento entre as classes apresetadas no GREN-Wizard com as classes da SiGCli

SiGCli			**GREN**			
Nº	***Padrão***	***Classes***	***Nº***	***Padrão***	***Variante***	***Classes***
1	Identificar Pacientes	Paciente	1	Identificar o Recurso	Múltiplos tipos de recursos	Recurso
		Informação Adicional	2	Quantificar o Recurso	Recurso Simples	Tipos de Recurso
2	Definir Serviços	Serviço	1	Identificar o Recurso	Default	Recurso
		Tipo do Serviço	2	Quantificar o Recurso	Recurso Simples	Tipos de Recurso
3	Realizar Vendas	Produto	1	Identificar o Recurso e	Default	Recurso
		Unidade de Medida	2	Quantificar o Recurso	Recurso Mensurável	Unidade de Medida
		Paciente	6	Comercializar o Recurso	Venda sem origem	Destino
		Venda				Comercialização do Recurso
		Serviço / Produto				Recurso
		Venda	11	Itemizar Transação do Recurso	Default	Transação do Recurso
		Item Vendido				Item de transação
		Serviço / Produto				Recurso
4	Processar Guias	Paciente	9	Manter Recurso	Default	Recurso

		Guia				Manutenção do Recurso
		Convênio				Origem
5	Agendar Atendimentos	Venda	14	Identificar Tarefas de Manutenção	Default	Comercialização do Recurso
		Guia				Manutenção do Recurso
		Agenda				Tarefa de Manutenção
6	Identificar Atendentes	Agenda	13	Identificar Executor da Transação	Default	Tarefa de Manutenção
		Atendente				Executor da Tarefa
7	Realizar Acompanhamento					
8	Realizar Compras	Fornecedor	6	Comercializar o Recurso	Default	Origem
		Compra				Comercialização do Recurso
		Produto				Recurso
		Compra	11	Itemizar Transação do Recurso	Default	Transação do Recurso
		Item Comprado				Item de transação
		Produto				Recurso
9	Controlar Faturamento	Venda	12	Pagar pela Transação do Recurso		Transação do Recurso
		Guia				Transação do Recurso
		Compra				Transação do Recurso
		Pagamento				Pagamento
		Taxa de Juros				Taxa de Juros
		Taxa de Multa				Taxa de Multa

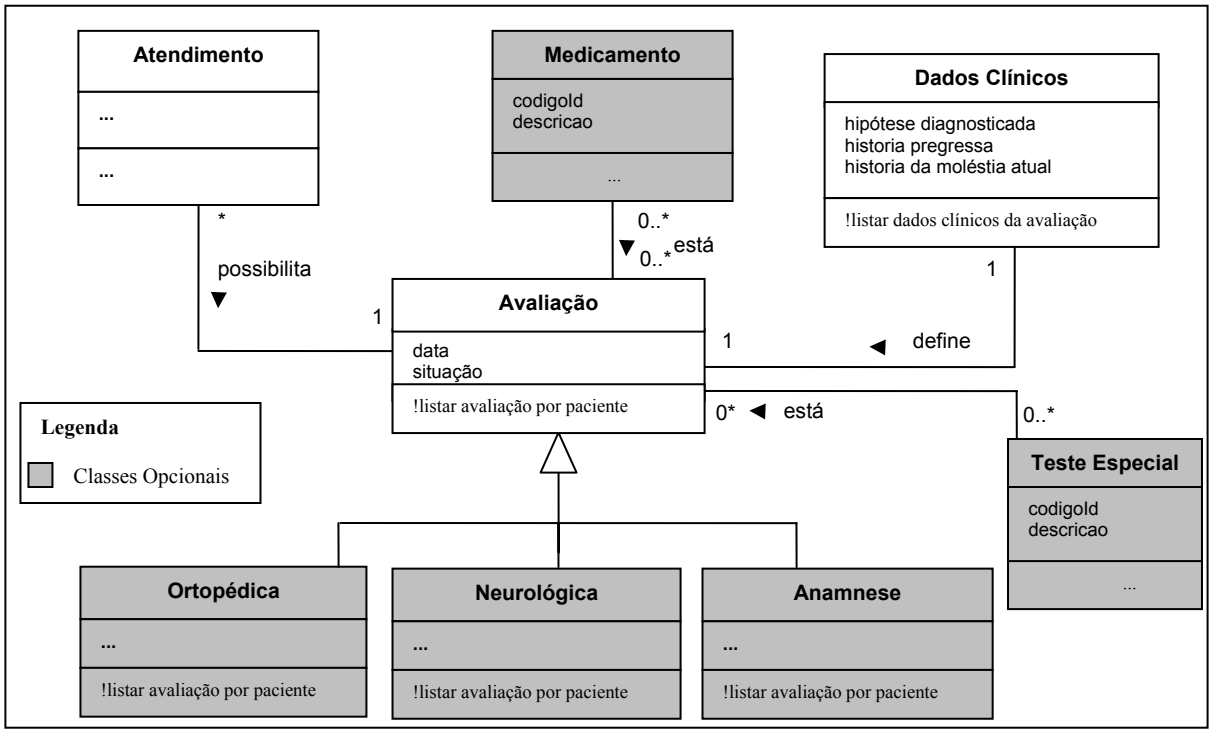

Figura 7 – Diagrama de classes do padrão 7 Realizar Acompanhamento da SiGCli.

Esse mapeamento, Tabela 3, facilita o uso do GREN-Wizard para instanciar aplicações no domínio SiGCli, uma vez que o desenvolvedor consegue identificar, de forma rápida, qual papel cada classe da SiGCli exerce em relação a sua possível aplicação no GREN-Wizard. A Figura 8 exibe a aplicação, no GREN-Wizard, do padrão 2 da SiGCli *Definir Serviços, que* é mapeado no padrão *Identificar Recurso*, da GRN, com a variação *Default,* como é definido na Tabela 3. A instanciação do padrão 2 da SiGCli acontece sem problemas, uma vez que todas as classes possuem mapeamento com as classes definidas para o Identificar Recursos da GRN.

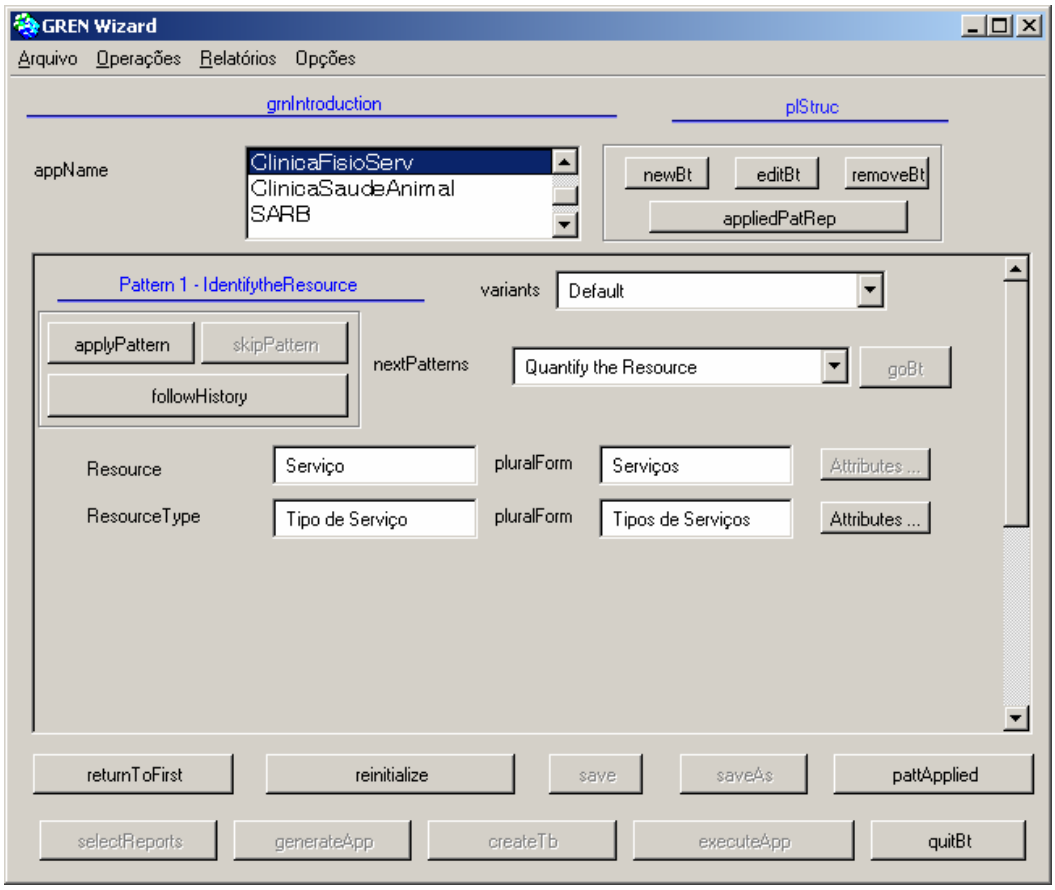

Figura 8 - Aplicando o padrão 2 da SiGCli, *Definir Serviços*, no GREN-Wizard

As classes que não possuem mapeamento devem ser instanciadas através de um processo manual, sem a ajuda do Wizard, na qual deve-se investigar se existem classes no *framework* GREN que podem servir como *template* para as novas classes. Para essa implementação o desenvolvedor deve possuir conhecimentos sobre programação em linguagem *Smalltalk* e estudar a estrutura interna do *framework* para que essas classes possam utilizar os recursos disponíveis pelo *framework*. Todas as adaptações sofridas pelo framework devem ser documentadas para um futuro aprimoramento da linguagem de padrões GRN e do GREN-Wizard. A Figura 7 apresenta o diagrama de classes do padrão 7 da SiGCli, *Realizar Acompanhamentos*, que não é coberto pelo *framework.*

5. Considerações Finais

O GREN-Wizard é uma ferramenta desenvolvida em *Smalltalk* que foi elaborada para facilitar a instanciação de aplicações que venham a usar o *framework* GREN e são cobertas pelo domínio da linguagem de padrões GRN. Ela possui uma interface amigável que guia o desenvolvedor durante o processo de instanciação da aplicação. A única exigência da ferramenta é que o desenvolvedor conheça a GRN e saiba quais padrões devem ser aplicados para obter a aplicação desejada.

Durante a elaboração da SiGCli, procurou-se sempre buscar padrões associados aos da GRN, para facilitar uma futura instanciação usando o GREN-Wizard. Entretanto, algumas funcionalidades específicas da SiGCli não são cobertas por esses padrões. Dessa forma, foram feitas adaptações aos padrões da GRN sempre que necessário. Após a elaboração dos padrões que definem a SiGCli, foram realizados três estudos de caso com o objetivo de minimizar os esforços na instanciação das aplicações. Para isso, os três sistemas usados no processo de análise de domínio foram remodelados, seguindo os padrões propostos pela SiGCli e foram aplicados no GREN-Wizard.

Durante o processo de instanciação, usando o GREN-Wizard, foram identificados dois fatores importantes que dificultam a instanciação de aplicações: a) a definição dos papeis que cada classe exerce nos padrões mapeados. Por exemplo, a classe *Paciente* assume dois papéis distintos durante a instanciação da aplicação, ora sendo o recurso ora sendo o destino da comercialização de um produto, outras classes também sofrem esse tipo de mutação. Tais mudanças de papéis não são cobertas pelo *framework* GREN, o que dificulta o entendimento do desenvolvedor para instanciar a aplicação; b) as funcionalidades específicas do domínio da SiGCli que não são previstas pelo *framework* exigindo do desenvolvedor conhecimento profundo do *framework* GREN e da linguagem *SmallTalk* para implementá-los.

Sendo o interesse por sistemas implementados em linguagem Java com interface Web usando a arquitetura três camadas e *Smalltalk* uma linguagem bem difundida no meio científico, mas sem grande representatividade no mercado, os esforços para a alteração do *framework* GREN são grandes e os resultados obtidos não seriam satisfatórios as necessidades reais das aplicações. Um novo padrão deveria ser incorporado à linguagem de padrões GRN, o padrão 7 da SiGCli, e implementado em *Smalltalk* para ser adicionado à estrutura do *framework* de modo que essa arquitetura não fosse violada. Dessa forma, alterações seriam necessárias nas camadas: *GREN-Negócio*, *GREN-Interface Gráfica com o Usuário* e *GREN-Wizard* do *framework* GREN.

Devido a não viabilidade da utilização do *framework* GREN com a linguagem de padrões SiGCli optou-se pelo desenvolvimento de um gerador de aplicações específico para esse domínio, apoiado nessa linguagem de padrões, produzindo aplicações Java com interface Web em arquitetura três camadas [7].

Referências

[1] BRAGA, R. T. V.; MASIERO, P. C. A process for framework construction based on a pattern language. In: *26th Annual International Computer Software and Applications Conference (COMPSAC 2002)*, Oxford–England, to appear, 2002.

[2] RÉ, R. *Um processo para construção de frameworks a partir da engenharia reversa de sistemas de informação baseados na Web: Aplicação ao domínio dos leilões virtuais*. Dissertação de Mestrado, ICMC/USP, São Carlos – SP, 2002.

[3] FAYAD, M. E.; E.JOHNSON, R.; SCHMIDT, D. C. *Building application frameworks: Objectoriented foundations of framework design*. John Wiley & Sons, 1999.

[4] COPLIEN, J. O. *Software design patterns: Common questions and answers* in L. Rising – The Patterns Handbook: Techniques, Strategies, and Applications, Cambridge University Press, p. 311–320, 1998.

[5] JOHNSON, R. E. Frameworks = (components + patterns). *Communications of the ACM*, v. 40, n. 10, p. 39–42, 1997b.

[6] JOHNSON, R. E.; RUSSO, V. *Reusing object-oriented designs*. Rel. T´ec. UIUCDCS 91-1696, University of Illinois, 1991.

[7] PAZIN, A. Um Gerador de Aplicações para o Domínio de Clínicas de Reabilitação. Dissertação de Mestrado a ser apresentada – Programa de Pós Graduação em Ciência da Computação – Universidade Federal de São Carlos, São Carlos – SP, agosto/2004

[8] BRAGA, R. T. V.; GERMANO, F. S. R.; MASIERO, P. C. A family of patterns for business resource management. In: *5th Annual Conference on Pattern Languages of Programs (PLOP'98)*, Washington University in St. Louis – Missouri, USA, on-Line. Disponível em http://jerry.cs.uiuc.edu/~plop/plop98/final_submissions. Consultado em 31/01/2003., 1998.

[9] BRAGA, R. T. V.; GERMANO, F. S. R.; MASIERO, P. C. A pattern language for business resource management. In: *6th Pattern Languages of Programs Conference (PLoP'99)*, Monticello – IL, USA, 1999.

[10] BRAGA, R. T. V. Um Processo para a Construção e Instanciação de *Frameworks* baseado em uma Linguagem de Padrões para um Domínio Específico. Tese de Doutorado, ICMC/USP, São Carlos-SP, 2003.

[11] BROWN, A. F. Como desenvolver aplicações com o SGBD ZIM. Rio de Janeiro: LTC – Livros Técnicos e Científicos Editora, 1990. 297p.

[12] ROSE - Visual Modeling with Rational Rose Home. Página com mais informações disponível na URL:http://www.rational.com/products/rose/index.jsp. Último acesso em 16 de Abril de 2003.

[13] ROBERTS, D.; JOHNSON, R. *Evolving frameworks: A pattern language for developing object oriented frameworks* in Martin, R.C., Riehle, D. , Buschmann, F. Pattern Languages of Program Design 3, Addison-Wesley, p. 471–486, 1998.