Utilização de Padrões de Projeto de Software na Reengenharia de Sistemas

Gustavo A. Prieto *
Rosângela D. Penteado

Universidade Federal de São Carlos – UFSCar
Departamento de Computação
Caixa Postal 676
13565-905 – São Carlos – Sp- Brasil

{prieto, rosangel}@dc.ufscar.br

Resumo

Este trabalho visa dar continuidade à implementação do sistema StatSim (*Statecharts Simulator*), em linguagem orientada a objetos e com utilização de padrões de software em linguagem Java e banco de dados relacional *Sybase*. Dessa forma os modelos recuperados na engenharia reversa e o projeto realizado na reengenharia parcial, já efetuada, serão utilizados. A continuidade do processo de reengenharia do ambiente StatSim ocorrerá com a inclusão de uma interface gráfica para apoiar os recursos de edição e simulação de *Statecharts* existentes na versão anterior. Alguns padrões propostos por Gamma e Grand e o padrão de projeto *Model-View-Controller* serão estudados e integrados ao ambiente na medida do possível. Um experimento será conduzido para verificar a manutenibilidade das versões existentes desse ambiente.

Palavras-chave:

Reengenharia, padrões de projeto de software, engenharia reversa, manutenção, desenvolvimento orientado a objetos.

---

* Trabalho realizado com o apoio financeiro da CAPES.

1. Introdução

A manutenção de sistemas [8] é a fase do ciclo de vida que mais absorve investimentos e esforços dentro das organizações. Fatores como falta de documentação, falta de organização do processo de desenvolvimento e mudança constante de tecnologias, plataformas e ferramentas vêm agravar esse quadro de forma alarmante. Uma das metas primordiais da engenharia de software é facilitar a realização de mudanças e diminuir o esforço de manutenção.

Denominam-se sistemas legados, sistemas que estão em funcionamento há vários anos e que atendem aos requisitos dos seus usuários, desempenhando funções críticas. Esses sistemas costumam consumir muitos recursos para manutenção. Muitas vezes há necessidade de expandi-los, alterar sua interface e, até mesmo, mudar sua linguagem de implementação para outra mais atual. O processo de engenharia reversa pode ser utilizado para recuperar um modelo com alto grau de abstração desse sistema e posteriormente o mesmo pode ser implementado utilizando recursos mais atuais, processo conhecido por reengenharia.

Com o advento da orientação a objetos, cogita-se freqüentemente a alteração do paradigma orientado a procedimentos, dos sistemas legados, para o orientado a objetos. Existem vários métodos, técnicas e abordagens para auxiliar na atividade de reengenharia e engenharia reversa. Entre elas, a abordagem Fusion/RE [4, 5, 6] que foi utilizada no ambiente StatSim que é alvo deste trabalho.

As técnicas para a solução de problemas de projeto, apresentadas de forma essencial, que podem ser utilizadas de forma recorrente toda vez que problemas similares surgirem são conhecidas como padrões de projeto [3, 7, 9]. Os padrões apresentam como vantagem o aumento de produtividade do desenvolvimento de software, uniformidade na estrutura do software, incremento da padronização no desenvolvimento, aplicação imediata por outros desenvolvedores e redução da complexidade do sistema. Com a utilização de padrões é possível a elaboração de sistemas mais flexíveis e funcionais, com maior reutilização e maior qualidade. Assim, o tempo de desenvolvimento, o esforço de implementação e a manutenção do sistema é reduzido.

2. Proposta do Projeto de Pesquisa

O ambiente StatSim, para edição e simulação de Statecharts [10], originalmente desenvolvido em C com orientação a procedimentos e utilizando arquivos texto para armazenamento dos dados, passou pelo processo de engenharia reversa usando o Fusion/RE [5], com o qual foi elaborado seu modelo de análise orientado a objetos. Continuando as pesquisas, o ambiente StatSim teve o seu código em C segmentado [6] e passou por reengenharia parcial [1, 2] com utilização do padrão de projeto Persistence Layer [7], linguagem Java e banco de dados relacional Sybase. A partir desses trabalhos nota-se um aumento na manutenibilidade do sistema, principalmente, maior facilidade no entendimento do mesmo.

Este projeto de pesquisa visa dar continuidade aos trabalhos de Cagnin [1, 2], com o objetivo de encontrar e implementar outros padrões que possam ser utilizados na reengenharia do ambiente StatSim em linguagem Java [12] e com banco de dados relacional Sybase [11].

Gamma e outros [3] propõem o uso de padrões de projeto de software como um novo mecanismo para expressar soluções na elaboração de projetos orientados a objetos. Com os estudos realizados até o momento, os padrões *Memento*, *Visitor*, *Observer* e *Proxy*, apresentam-se como candidatos a serem utilizados na implementação orientada a objetos.

O padrão *Model-View-Controller* [13] tem como meta separar o processamento da

informação (funcionalidade) de sua representação (interface). Seu objetivo é o de facilitar a resolução dos problemas que surgem quando usuários necessitam de mudança da interface, plataforma ou a criação de aplicações com interfaces diferentes (usuário, gerente). Esse padrão aumenta a flexibilidade, portabilidade e a reusabilidade pois divide a aplicação em três componentes: *model* que contém o núcleo dos dados e da funcionalidade; *controller* que manipula a entrada dos usuários (cliques do mouse, interações de menus, etc); *view* que apresenta as informações aos usuários. Um mecanismo de mudança/atualização através do qual um componente *model* difunde uma notificação de mudança para todos os seus dependentes mantendo-os sempre atualizados, também é definido.

Desenvolvedores de sistemas orientados a objetos que se utilizam de bancos de dados relacionais perdem muito tempo e esforço na implementação de mecanismos que possibilitem tornar persistentes os objetos. Nesses sistemas coexistem dois paradigmas com diferenças claras: os objetos consistem de dados e comportamentos e possuem herança; enquanto que bancos de dados relacionais são constituídos de tabelas, colunas, relações e funções de cálculos básicos. Uma solução é a implementação do padrão *Persistence Layer* [7], que viabiliza uma camada que cuida da interface entre os objetos e o banco de dados, e assim, protege ambos de mudanças constantes.

3. Estágio do Trabalho

A figura 1 ilustra o trabalho a ser desenvolvido. Os retângulos de bordas arredondadas representam o material disponível enquanto que os de bordas retas, os resultados que se pretende obter quando da conclusão deste trabalho.

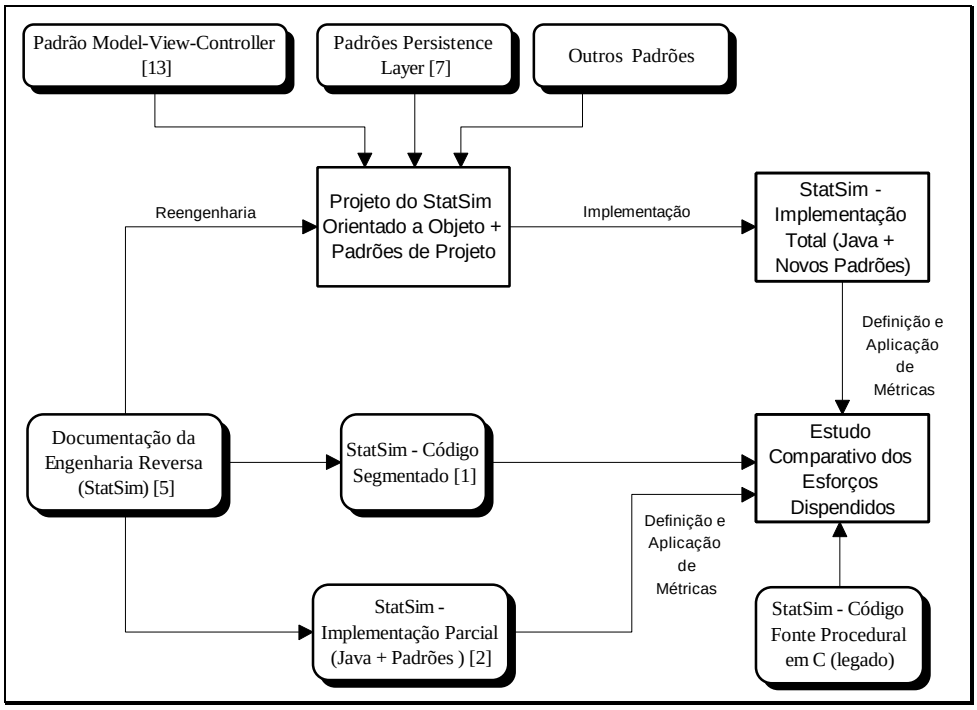

Figura 1 - Trabalho a ser Desenvolvido

Quando do início deste trabalho, estava disponível ao mestrando: 1) a versão original do software legado implementado em linguagem procedimental; 2) a documentação de análise dessa versão obtida através da engenharia reversa, utilizando a abordagem *Fusion/RE*; 3) o código segmentado obtido após a engenharia reversa do ambiente StatSim, em que foi mantida a funcionalidade e a linguagem de implementação sendo o paradigma de

desenvolvimento orientado a objetos; 4) implementação parcial do ambiente StatSim em linguagem orientada a objetos (Java) com uso de padrões de projeto.

É intenção deste trabalho além de completar a reengenharia do ambiente StatSim e efetuar um experimento controlado para avaliar a manutenibilidade do sistema legado e do sistema reengenheirado [6]. Esse experimento será conduzido de forma a comparar manutenibilidade do sistema desenvolvido com a do sistema legado.

Vários padrões foram estudados até o momento como: os padrões de projeto que compõem o Persistence Layer [7], o Model-View-Controller [13], e alguns dos apresentados por Gamma e outros [3, 9]. A implementação de uma interface gráfica programada em Java, que integrará toda a funcionalidade do StatSim com os padrões citados, teve início.

O projeto de pesquisa quando concluído constará do ambiente StatSim reengenheirado em linguagem orientada a objeto, Java, utilizando padrões de projeto.

Referências Bibliográficas

[1] **Cagnin, M. I.; Penteado, R. A. D.; Germano, F. R. S.; Masiero, P. C.** – Reengenharia com uso de Padrões de Projeto. XIII Simpósio Brasileiro de Engenharia de Software, pág. 273-288. 1999

[2] **Cagnin, M. I. –** Avaliação das Vantagens quanto à Facilidade de Manutenção e Expansão de Sistemas Legados Sujeitos a Engenharia Reversa e Segmentação. Dissertação de Mestrado – Departamento de Computação, Universidade Federal de São Carlos. São Carlos. 1999

[3] **Gamma, E., Helm, R., Johnson, R., Vlissides, J.** – *Design Patterns – Elements of Reusable of Object Oriented Software*, Addison-Wesley. 1995.

[4] **Penteado, R. A. D.; Masiero, P. C.; Braga, R. T. V. –** Improving the Quality of Legacy Code by Reverse Engineering. Proceedings of 4th International Conference on Information Systems, Analisys and Sinthesis, ISAS'98, Orlando – Florida, p. 364-370. 1998.

[5] **Penteado, R. A. D.; Masiero, P. C.; Germano, F. S. R. –** An Overall Process Based on Fusion to Reverse Engineer Legacy Code. Proceedings of 3rd Working Conference on Reverse Engineering, Monterrey-California. IEEE, p. 179-188. 1996.

[6] **Penteado, R. A. D.; Masiero, P. C.; Cagnin, M. I. –** An Experiment of Legacy Code Segmentation to Improve Mainteinability. Proceedings of 3rd European Conference on Software Maintenance and Reengineering, CSMR'99, Amsterdan – The Netherlands. IEEE, p. 91-100. 1999.

[7] **Yoder, J. W.; Johnson, R. E.; Wilson, Q. D.** – Connection Busines Objects to Relational Databases. Proceeding of 5th Conference on the Patern Languages of Programs, Monticello-IL, EUA. Proceedings, 1998.

[8] **Pressman, R. S.** – *Engenharia de Software*, Makron Books, p. 876-914. 1995.

[9] **Grand, M.** – Patterns in Java – A Catalog of Reusable Design Patterns Illustrated with UML. Volume 1. Wiley Computer Publishing. 1998.

[10] **Harel, D**. – STATECHARTS: A visual formalism to Complex Systems. Science of Computer Programming, v. 8, p. 231-274. 1987.

[11] **Sybase -** Sybase Inc. URL: http://www.sybase.com.

[12] **Java** – JDBC Basics. URL: http://www.java.sun.com.

[13] **Buschmann, F.; Meunier, R.; Robert, H.; Sommerland, P.; Stal, M.** – *Pattern – Oriented Software Architecture: A System of Patterns*. P. 125-143.