Aspectos de Validação do Método de Engenharia Reversa Fusion-RE/I aplicado a um Sistema Hipermídia

Valéria Delisandra Feltrim[1]
Renata Pontin de M. Fortes
Willian Francisco da Silva
{vfeltrim, renata, wfsilva}@icmc.sc.usp.br

Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo
C.P. 668 13560-970 São Carlos - São Paulo - Brasil

Abstract

The growth of the software market has leading to an increasing use of development techniques, which are, sometimes, informal ones. The maintenance of such software is problematic, since its documentation rarely reflects the implemented code. In this context Reverse Engineering of Software can help by means of recovering the project information lost during the development phase and documenting the current software state. This article discusses the issues emerged during the application of the method of reverse engineering named Fusion-RE/I. The described experiment is part of the re-engineering of a prototype hypermedia system, which has, as main goal, to adapt it to a Software Engineering domain. Since the target is a hypermedia system, the results obtained during the use of Fusion-RE/I can be registered as a hyperdocument. By doing that, it is possible to observe and analyse some relevant issues concerning the method Fusion-RE/I.

1. Introdução

O crescimento do mercado de software a cada dia acarreta o aumento do uso de técnicas de desenvolvimento, muitas vezes informais. Assim, a documentação associada ao software, na maioria das vezes, é negligenciada e não está de acordo com o código implementado [1]. Além disso, as constantes modificações e adições de novas características ao software acarretam efeitos colaterais inesperados, que não estão presentes na documentação.

Dessa forma, quando diante da manutenção do produto, o engenheiro de software encontra uma documentação informal e incompleta, que não reflete o software existente. Sendo assim, a manutenção de software se constitui uma fase reconhecidamente problemática, responsável por custos de proporções superiores aos das demais fases do ciclo de vida de sistemas [2].

A atividade de manutenção consiste de três etapas: entendimento, modificação e revalidação do sistema [3]. Notadamente, as etapas de entendimento e modificação estão muito relacionadas com a disponibilização das informações do software, ou seja, se apóiam na existência, consistência, completitude e atualização correta dos documentos que o compõem.

A engenharia reversa procura contribuir para o aumento da manutenibilidade do software, fornecendo informações, primeiramente utilizadas no entendimento e posteriormente na modificação e revalidação do software [4]. Isso se dá através da recuperação de informações a

[1] Este trabalho tem o apoio financeiro da FAPESP, #97/12999-8.

partir dos documentos do software, visando obter sua representação em um nível mais alto de abstração e de forma mais clara.

Por sua vez, o processo de engenharia reversa, de forma geral, não é trivial e exige um alto custo pessoa/tempo para sua realização. Isso devido ao grande volume de informações envolvido durante o desenvolvimento do processo. Além disso, manter a consistência dos relacionamentos existentes entre as informações recuperadas é crucial para que o processo seja eficiente e permita um melhor entendimento do software sob análise.

A tecnologia de hipertextos, servindo como um meio mais flexível para tornar disponível as informações aos usuários de aplicativos, está evoluindo rapidamente, possibilitando que novas formas de recuperação e apresentação das informações também sejam investigadas. De fato, a base da tecnologia de sistemas hipertexto é a rede de informações que possui interconexões que devem estar facilmente acessíveis pelos usuários[2]. Essa rede de informações, onde os "nós" correspondem às unidades de informação e os "*links*", às interconexões entre eles, compõe um hiperdocumento. Na engenharia de software, além de um grande volume de documentos, existe uma grande diversidade de tipos de documentos (diagramas, textos, códigos-fonte, executáveis, etc.). Tal característica, aliada ao fato de que tais documentos contêm informações bastante relacionadas, sugere naturalmente a utilização de hiperdocumentos como meio adequado para o armazenamento e recuperação dessas informações [5].

Um dos interesses deste trabalho está no tratamento de hiperdocumentos para dar o suporte necessário ao processo de engenharia reversa de software. Dessa forma, foi elaborada uma especificação da estrutura de hiperdocumento que possibilitasse a fidelidade do conteúdo da documentação com relação ao produto de software sendo documentado, durante a aplicação do método de engenharia reversa. Além disso, almeja-se que os relacionamentos entre as informações do hiperdocumento e os componentes do software estejam acessíveis com mais facilidade por meio dos *links* definidos.

Para tanto, um ambiente de hipertexto, SASHE (*S*istema de *A*utoria e *S*uporte *H*ipermídia para *E*nsino) [6], foi submetido à engenharia reversa, e as etapas de tal processo, posteriormente registradas no próprio SASHE. Cada etapa de recuperação dos níveis de abstração e das diversas relações entre os componentes de software implementados no SASHE, corresponde à definição de nós e *links* que o documentam, de forma hipertextual, no próprio SASHE.

Esse ambiente de hipertexto, SASHE, foi desenvolvido em um projeto anteriormente concluído, e a partir dos primeiros experimentos obtidos com a criação de hiperdocumentos para ensino, pode-se observar que suas características constituem um recurso potencialmente utilizável por outros tipos de aplicação, que não somente as de ensino [7], [8]. Neste sentido, a possível generalização para outro domínio de aplicação, o de engenharia reversa de software, foi nossa principal motivação para o início do trabalho aqui relatado. Complementando o que já vem sendo desenvolvido, será realizada a etapa final da re-engenharia do sistema SASHE, de modo a torná-lo adequado ao domínio de Engenharia de Software.

A utilização do SASHE para "se autodocumentar" durante o processo de engenharia reversa a que foi submetido foi, então, o exercício fundamental para o levantamento de requisitos que um engenheiro de software necessita para utilizar o SASHE como ferramenta de suporte à documentação do processo de engenharia reversa de software. Além disso, esse exercício propiciou uma avaliação interativa dos resultados obtidos durante a realização da engenharia reversa.

---

[2] O usuário de sistemas hipertexto pode ser: leitor e autor.

Optou-se pela utilização do método Fusion-RE/I [9] devido, principalmente, ao fato de que a experiência prática que foi conduzida se constituiu da aplicação da engenharia reversa a um ambiente de hipertexto, e esta aplicação, por sua vez, registrada no próprio ambiente hipertexto. Como a utilização do Fusion-RE/I inicialmente considera a operacionalidade da interface usuário-computador do software, foi possível uma intensa interação para a própria "autodocumentação" realizada.

Além disso, o método Fusion-RE/I foi escolhido também pelo fato que o mesmo leva à produção das visões do sistema, ou seja, auxilia no entendimento e reconhecimento do modelo abstrato do software, sob o paradigma de Orientação a Objetos (OO).

Estando praticamente terminada a tarefa de engenharia reversa com base no método Fusion-RE/I, no ambiente SASHE, foi possível observar pontos positivos e pontos críticos na aplicação desse método, baseados na experiência adquirida durante a execução do trabalho.

Na próxima seção são apresentados o método Fusion-RE/I [9] e os produtos gerados em cada uma de suas etapas. A Seção 3 mostra uma modelagem conceitual do domínio de informações de engenharia reversa, tendo como base as premissas do método Fusion-RE/I. Pelo resultado final do processo tratar-se de um hiperdocumento, essa modelagem foi feita utilizando-se o método de modelagem hipermídia OOHDM [10], [11], [12]. Na Seção 4 são apresentados os produtos gerados durante a aplicação da engenharia reversa e também algumas métricas relativas a esse processo. As considerações feitas sobre a aplicação do método Fusion-RE/I, como forma de validação do método, puderam ser obtidas e são descritas na Seção 5. Finalmente, na Seção 6 apresentamos as conclusões do trabalho realizado.

2. O método utilizado: Fusion-RE/I

O método Fusion-RE/I constitui-se de um método de engenharia reversa que, visando facilitar o processo, parte da interface do sistema para a recuperação de informações úteis ao entendimento do software. Através desse método é possível a recuperação de visões funcionais e de visões estruturais do sistema. As considerações lógicas são obtidas por meio da análise da interface para a recuperação de visões funcionais do sistema e, posteriormente, parte-se dessas visões recuperadas e de considerações físicas obtidas por meio da análise do código fonte, para a recuperação de visões estruturais do sistema [9]. O método Fusion-RE/I utiliza os modelos da fase de análise do método de desenvolvimento de software orientado a objetos Fusion [13] para representar as informações recuperadas. A Figura 1 mostra uma visão geral do método.

O método Fusion-RE/I surgiu a partir de outro método de engenharia reversa, o Fusion-RE [14]. O Fusion-RE/I considera a mesma abordagem de seu predecessor, o Fusion-RE, de recuperar os modelos de análise Fusion. Porém possui características distintas quanto a ordem das etapas a serem cumpridas, e, o processo do Fusion-RE/I inicia-se com a análise da interface do sistema para a recuperação das visões [9]. A seguir, são descritas resumidamente as duas etapas do método Fusion-RE/I, com suas tarefas. No final da seção é apresentado um quadro (Quadro 1) com um resumo das etapas prescritas pelo método.

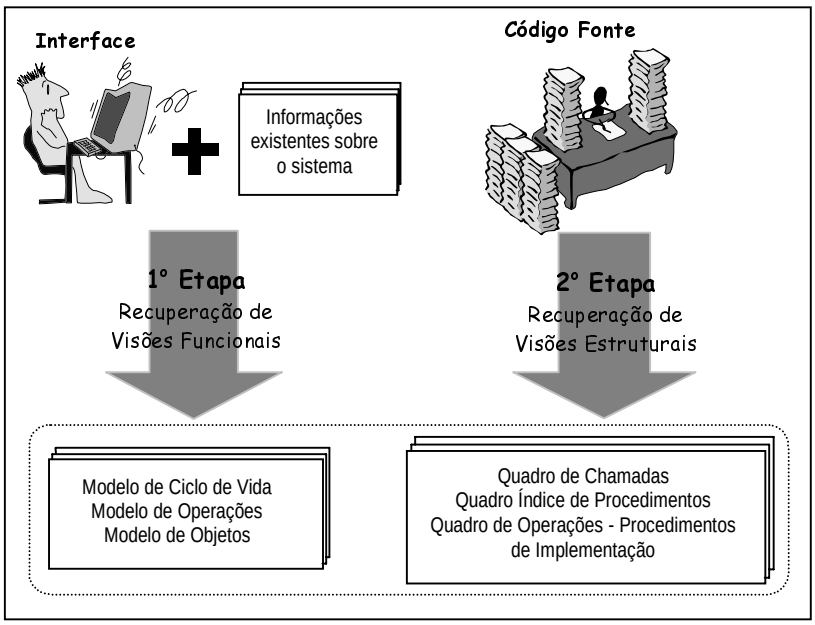

Figura 1. Visão Geral do Método Fusion-RE/I

2.1. Recuperação de Visões Funcionais - Etapa 1

Visando obter a abstração da funcionalidade do sistema, numa primeira etapa do método Fusion-RE/I, são realizadas duas tarefas: a obtenção de informações existentes sobre o sistema e a recuperação do modelo de análise do sistema, as quais são descritas a seguir.

Etapa1-a. Obter informações existentes sobre o sistema: busca-se toda a informação disponível sobre o sistema em estudo. Isso envolve reunir a documentação existente (manuais, listagens de código, etc.) sobre o sistema e também obter outras informações relevantes, como o domínio do sistema, linguagem de implementação, etc.

Entrevistas com os usuários também podem ser úteis já que, muitas vezes, informações importantes podem não estar documentadas.

Tendo reunida toda a documentação existente, a mesma deve ser analisada para identificação de informações relacionadas aos requisitos do sistema, ao projeto arquitetural, de dados e procedimental, ao ambiente onde o sistema é executado, a organização dos arquivos em disco, etc.

Etapa1-b. Recuperar Modelo de Análise do Sistema: após obtidas todas as informações existentes sobre o sistema, parte-se para a tarefa seguinte, de recuperação das informações da fase de análise. Todas as informações recuperadas nesse passo são obtidas por meio da investigação sobre a interface do sistema.

A tarefa de recuperação do modelo de análise do sistema, segundo [9], compreende a elaboração dos três modelos da fase de análise do método Fusion: o modelo de ciclo de vida, o modelo de operações e o modelo de objetos. Pressupõe-se que essa tarefa exige grande esforço, em função da complexidade do sistema. A seguir, são brevemente descritos os procedimentos para a elaboração dos modelos da fase da análise.

Etapa1-b1. Elaborar o Modelo de Ciclo de Vida do Sistema: a partir do uso do sistema, do estudo da documentação existente e das entrevistas com os usuários pode-se definir a seqüência de operações permitidas e os eventos de entrada e de saída que o sistema aceita.

As opções presentes no menu da interface do sistema formam a expressão principal do modelo de ciclo de vida. A partir das opções listadas na expressão principal, são construídas

novas expressões, uma para cada opção, mostrando as seqüências de operações permitidas a partir daquele ponto. Para cada operação citada, é escrita uma nova expressão identificando a seqüência permitida de eventos de entrada (elementos da operação) e os respectivos eventos de saída, os quais aparecem precedidos pelo símbolo #.

Etapa1-b2. Elaborar o Modelo de Operações do Sistema: parte-se do modelo de ciclo de vida obtido anteriormente. Esse modelo apresenta uma visão geral da funcionalidade das operações do sistema, as quais, para a elaboração do modelo de operações devem ser melhor estudadas, através do uso intensivo do sistema, de modo que possam ser especificadas detalhadamente.

Dessa forma, descreve-se a execução de cada operação partindo-se do modelo de ciclo de vida anterior. O método Fusion-RE/I considera não apenas as operações e eventos gerados através da interface, mas também operações e eventos não visíveis em tela, como criação e manipulação de arquivos. Isso se dá através da observação do "diretório de trabalho" do sistema, após a execução de cada operação da interface, verificando a modificação ou a criação de novos arquivos.

Uma vez compreendida a operação, pode ser preenchido o formulário de descrição da mesma. Deste formulário constam os seguintes itens: `nome`, `descrição`, `lê`, `modifica`, `envia`, `assume` e `resultado`.

A `descrição` da operação pode ser obtida através do manual do usuário. As informações do item `lê` são obtidas a partir da observação da interface, identificando-se o valor referente a cada elemento acessado pela operação (eventos de entrada). Através da observação da interface e da execução da operação obtém-se as informações do item `modifica`, identificando-se o valor referente a cada elemento acessado e modificado pela operação. As informações do item `envia` são obtidas através da observação da interface, identificando os eventos de saída gerados pela operação. As informações do item `assume` especificam as pré-condições necessárias para a execução da operação e também são obtidas pela observação da interface. As informações do item `resultado` podem ser preenchidas identificando-se, através da execução da operação, o relacionamento entre o estado inicial do sistema (antes da operação) e o estado final do sistema (após o término da operação).

Etapa1-b3. Elaborar o Modelo de Objetos do Sistema: no Fusion-RE/I primeiramente definem-se os assuntos relacionados com a funcionalidade do sistema. Esses assuntos são denominados *Temas*. Para que se possa definir os temas é necessária uma análise das informações recuperadas nos passos anteriores e também das abstrações registradas nos modelos de ciclo de vida e de operações. Essa é uma das tarefas mais subjetivas do método Fusion-RE/I e é de fundamental importância para a aplicação do método [9].

Tendo definido os temas, é feito um agrupamento das operações de acordo com os temas a que se referem. Assim, ao final tem-se uma lista de temas e as operações dos temas. Temas podem ser constituídos de outros temas, quando os assuntos se relacionam em um nível de abstração mais alto.

Para cada um dos temas definidos, é construído um modelo de objetos. Para isso, as operações são novamente analisadas, agora buscando identificar componentes que constituem o modelo de objetos, ou seja, classes, relacionamentos, atributos, e possíveis agregações, especializações e generalizações.

Os componentes do modelo de objetos podem incluir componentes que não estão explícitos nas operações, mas que são identificados pela abstração e entendimento da funcionalidade de cada elemento e de cada operação.

A construção do modelo de objetos é uma tarefa bastante subjetiva, e embora não conste no método, provavelmente vá requerer um *feedback* após a recuperação das visões estruturais (visualização do código). A necessidade desse feedback ficou então evidente durante a utilização do Fusion-RE/I. Uma vez que as principais referências sobre esse método de engenharia reversa não emitem uma diretriz única para a elaboração desse modelo de objetos, foi uma investigação importante, durante a aplicação do método, se identificar qual o melhor momento para sua elaboração. Em [9], o método Fusion-RE/I é descrito, mas não há uma definição clara de que o modelo de objetos possa ser construído através da observação da interface somente.

2.2. Recuperação de Visões Estruturais - Etapa 2

Nesta etapa trabalha-se com o código fonte do sistema, tendo como objetivo identificar os procedimentos que implementam as operações do sistema discriminadas na etapa anterior. A seguir, são descritos os procedimentos que devem ser realizados nesta etapa.

Etapa2-a. Elaborar Quadro de Procedimentos de Implementação: tem-se como objetivo identificar cada procedimento, sua funcionalidade e a seqüência de chamadas desse procedimento. Para tal, são utilizados dois quadros: um quadro de chamadas para cada arquivo de programa do sistema e um índice geral de procedimentos.

Etapa2-a1. Elaborar o Quadro de Chamadas: esse quadro deve ser elaborado para cada arquivo de código fonte do sistema, apresentando os procedimentos contidos no arquivo, suas respectivas funcionalidades e os procedimentos utilizados (chamados) e utilizadores (chamado por).

As informações de descrição (funcionalidade) dos procedimentos podem ser conseguidas a partir de comentários no código fonte. Os procedimentos chamados são obtidos pela análise do próprio código, enquanto que os procedimentos utilizadores (chamados por) vão sendo obtidos à medida que os quadros são elaborados.

Ao final, os procedimentos de cada quadro são re-arranjados em ordem alfabética.

Etapa2-a2. Elaborar o Quadro Índice de Procedimentos: esse quadro apresenta todos os procedimentos da implementação do sistema em ordem alfabética, com as respectivas localizações (arquivo e diretório). Para a elaboração desse quadro utilizam-se todos os quadros de chamadas obtidos anteriormente.

Etapa2-b. Elaborar Quadro das Operações - Procedimentos de Implementação: identificam-se os procedimentos que implementam as operações da interface e, de acordo com sua funcionalidade, os procedimentos são alocados à interface ou a um dos temas definidos anteriormente. São, então, identificados os *links* entre os documentos “Quadro de Operações”, da primeira etapa do método, com os respectivos códigos que os implementam.

Nas primeiras colunas do quadro são colocadas as opções do menu e as operações de cada opção (descrição da interface). Na próxima coluna são colocados os procedimentos que implementam cada operação, de acordo com a hierarquia de chamadas descrita no quadro de chamadas. As próximas colunas do quadro são utilizadas para alocar os procedimentos à interface ou a um dos temas definidos. Assim, tem-se uma coluna para “interface” e uma para cada tema definido.

O nível de profundidade com que os procedimentos são detalhados nesse quadro depende do interesse em questão [9].

A seguir, são resumidas no Quadro 1 as tarefas prescritas para cada etapa do método.

Quadro 1. Ordem das tarefas prescritas pelo Fusion-RE/I

Etapa 1. Recuperação de Visões Funcionais
a. Obtenção de informações existentes sobre o sistema
b. Recuperação do modelo de análise
b1. Elaboração do modelo de ciclo de vida
b2. Elaboração do modelo de operações
b3. Elaboração do modelo de objetos
Etapa 2. Recuperação de Visões Estruturais
a. Elaboração do Quadro de procedimentos de implementação
a1. Elaboração do quadro de chamadas
a2. Elaboração do quadro índice de procedimentos
b. Elaboração do quadro de operações - procedimentos de implementação

3. Modelagem do hiperdocumento de apoio ao Fusion-RE/I

Conforme citado em seção anterior, o processo de engenharia reversa executado no sistema SASHE constitui o exercício para revelar os requisitos para a definição de uma estrutura de hiperdocumento adequada à engenharia reversa. Como ponto de partida para a construção do hiperdocumento, julgamos necessária a modelagem conceitual do processo de engenharia reversa, através de um método de modelagem conceitual. Para tal, optamos pelo OOHDM (*Object Oriented Hypermedia Design Methodology*) [10], [11], [12].

A modelagem conceitual ou de domínio destina-se à compreensão do domínio problema e à construção de modelos adequados deste domínio [12], no caso do OOHDM, identificando objetos e organizando-os através dos relacionamentos identificados. Dessa forma, como resultado tivemos, além da modelagem conceitual do domínio alvo como um todo, a definição de quais objetos devem ser efetivamente navegados no hiperdocumento, e como esses objetos podem ser navegados, definindo quais links devem existir, bem como os contextos em que cada objeto pode ser apresentado.

O processo prescrito pelo método Fusion-RE/I (Quadro 1) foi a base para o estabelecimento das atividades envolvidas e dos modelos gerados ao final do processo de engenharia reversa, e isto é refletido no resultado final da modelagem, que é composta por um esquema conceitual (modelo de objetos), um esquema de classes navegacionais e um esquema de contexto [15]. A Figura 2 apresenta parte do esquema conceitual do processo do Fusion-RE/I, no qual pode-se ver as classes de documentos gerados como resultado do processo e os relacionamentos entre eles.

Todo o processo descrito na modelagem foi instanciado com a realização do processo de engenharia reversa. De fato, os resultados apresentados na Seção 4, a seguir, são instâncias de objetos das classes identificadas neste esquema (Figura 2).

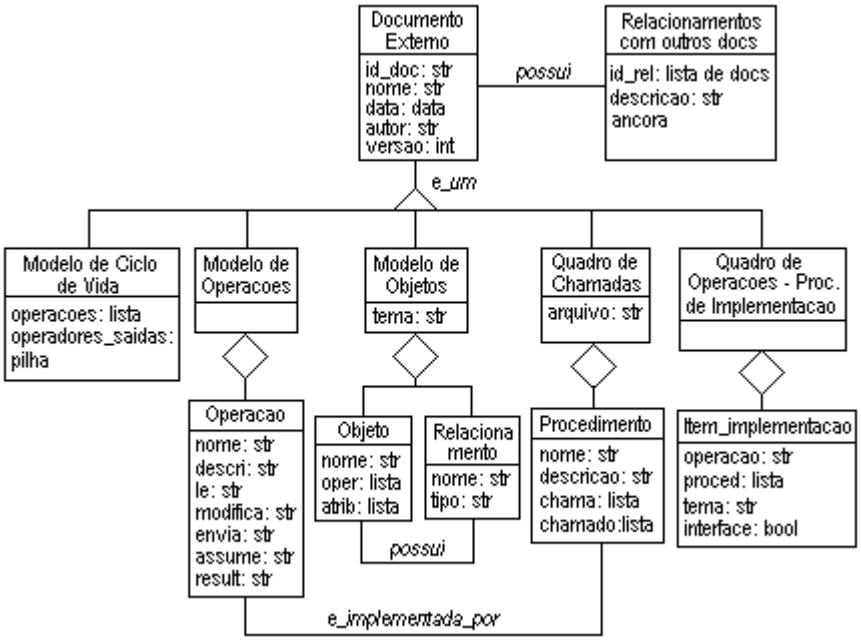

Figura 2. Esquema conceitual parcial do processo de engenharia reversa ditado pelo Fusion-RE/I

4. Resultados obtidos na utilização do Fusion-RE/I

Nesta seção são apresentados os produtos resultantes da aplicação do método Fusion-RE/I ao sistema SASHE, e também algumas medidas relativas aos produtos e ao processo desenvolvido. Com exceção da primeira tarefa especificada pelo método (obtenção de informações existentes sobre o sistema), todas as outras têm a elaboração de um documento como resultado final.

Como descrito na Seção 2, a primeira tarefa a ser realizada é a coleta de informações e documentação existentes sobre o sistema em questão (Etapa1-a). No caso específico do sistema SASHE, toda a informação disponível sobre o sistema resumia-se a dois relatórios técnicos [6], [16]. Importantes informações foram obtidas por meio de entrevistas com uma das pessoas que havia participado do desenvolvimento do sistema, e com usuários do sistema envolvidos em outros projetos relacionados ao SASHE.

A partir do uso intensivo do sistema, do estudo da documentação existente e das entrevistas com os usuários pôde-se definir a seqüência de operações permitidas e os eventos de entrada e de saída que o sistema aceita (Etapa1-b1). O modelo de ciclo de vida construído para o SASHE resultou em cerca de cinqüenta e cinco (55) sentenças compondo as seqüências de operações permitidas no sistema, incluindo ambos os módulos que compõem o software (Professor e Estudante).

Para facilitar a verificação das sentenças do ciclo de vida, foram construídos Diagramas de Transição de Estados (DTE) para cada sentença. O conjunto de DTE’s permitiu a verificação da corretitude das sentenças do ciclo de vida, uma vez que tornou mais clara a visualização de cada interação. Os DTE's foram feitos em uma ferramenta de apoio a modelagem de software, a Rational Rose [17]. Na Figura 3 é mostrada uma pequena parte (4 sentenças) do modelo de ciclo de vida construído.

```
life cycle SASHE = ( PROFESSOR | ESTUDANTE | Saída )

PROFESSOR = ( HIPERBASE | JANELAS | AJUDA | b_Criar_hiperbase |
b_ABRIR_HIPERBASE | b_SALVAR_HIPERBASE | b_ROTEIRO | b_GLOSSÁRIO | ELOS | NÓS )+

HIPERBASE = ( ((Criar | ABRIR) . SALVAR) | Sair )

ABRIR = ( (Nome_do_arquivo . (Pastas | Unidades_de_disco |
Listar_arquivos_do_tipo | Som._Leitura | b_REDE | b_AJUDA)* . ( (b_Ok .
(#msg_não_é_possível_encontrar_este_arquivo...). b_Ok . ABRIR) | (b_OK .
(#msg_este_nome_de_arquivo_tem_muitas_letras) . b_Ok . ABRIR) | b_Ok |
b_Cancelar ) ) | b_Cancelar )
                                        ...
```

Figura 3. Parte do Modelo de Ciclo de Vida construído para o sistema SASHE

O número elevado de sentenças desse modelo é um indicativo da grande interatividade do software e dos muitos estilos de interface (menus, botões, múltiplas janelas) implementados no sistema.

O modelo de operações é decorrente do modelo de ciclo de vida. Cada operação identificada no modelo de ciclo de vida é descrita individualmente através de um formulário textual, conforme descrito na Seção 2. Foram identificadas no modelo de ciclo de vida construído para o sistema SASHE sessenta e duas (62) operações, sendo que cada uma delas foi estudada exaustivamente para que pudesse ser descrita de acordo com os requisitos impostos pelo método (Etapa1-b2). Todos os formulários preenchidos, juntos, compõem o modelo de operações. A Figura 4 apresenta a descrição (um dos formulários) feita para a operação "Abrir" do menu "Hiperbase".

Operação:	Abrir
Descrição:	Abre uma hiperbase já existente
Lê:	Nome do arquivo; diretório; drive; somente leitura; botão ok; botão cancelar
Modifica:	
Envia:	Agente externo:{#msg não é possível encontrar este arquivo}; Agente externo:{#msg este nome de arquivo tem muitas letras}
Assume:	
Resultado:	Se botão cancela, operação é cancelada Se botão ok, Se arquivo existente, a hiperbase é aberta Se arquivo não encontrado, mensagem de arquivo não encontrado é enviada, e continua na operação Abrir hiperbase Se nome de arquivo tem mais de 8 letras, mensagem de nome com muitas letras é enviada, e continua na operação Abrir hiperbase

Figura 4. Descrição da operação "Abrir"

Para a elaboração do modelo de objetos (Etapa1-b3), primeiramente definiu-se os assuntos relacionados com a funcionalidade do sistema, assuntos esses denominados *Temas*. Para que se pudesse definir os temas, foi necessária uma análise das informações recuperadas na primeira tarefa (Etapa1-a) e também das abstrações vindas dos modelos de ciclo de vida e de operações. De fato, constatamos a subjetividade da definição dos temas, o que nos levou a várias tentativas de possíveis temas, antes de decidirmos quais seriam os temas definitivos do sistema.

A primeira tentativa de agrupamento das operações identificadas nos temas propostos revelou que aqueles temas definidos não eram adequados, pois algumas operações não se encaixavam em nenhum dos temas, e alguns temas não continham nenhuma operação. Além disso, os temas definidos não evoluíam para os modelos de objetos. Sendo assim, analisando

cuidadosamente as operações e seus relacionamentos, foram então propostos dois temas para o sistema: *Autoria* e *Navegação*, o que nos pareceu adequado. Foi feito então novo agrupamento de todas as operações identificadas no modelo de operações em um desses temas. As operações foram então novamente analisadas, buscando identificar componentes que constituíssem o modelo de objetos, ou seja, classes, relacionamentos, atributos, e possíveis agregações, especializações e generalizações. Dessa forma foram construídos dois modelos de objetos, um para o tema *Autoria* e um para o tema *Navegação*.

O modelo de objetos para o tema *Autoria* pode ser visto na Figura 5. É importante ressaltar que os modelos foram propostos baseados apenas na visão funcional do software, conseguida através da interação exaustiva com o software. Assim como com os DTE's, os modelos de objetos foram construídos na ferramenta Rational Rose [17].

Com o término da recuperação do modelo de ciclo de vida, do modelo de operações e dos modelos de objetos, encerra-se a primeira etapa do método Fusion-RE/I, que recupera visões funcionais do software unicamente através da interação com as informações sobre o sistema e sua interface. A conclusão dessa etapa do método deu-se ao final de cerca de 2 meses.

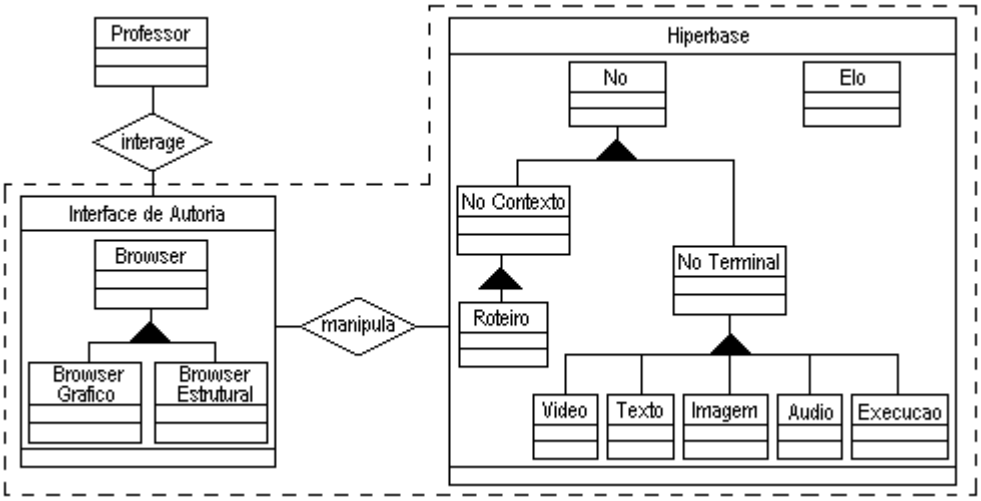

Figura 5. Modelo de objetos simplificado para o tema *Autoria*

Essa primeira etapa do método Fusion-RE/I lida com o sistema alvo como “caixa-preta”. Dessa forma, o suporte de registro em hiperdocumento para o acompanhamento da evolução das visões funcionais recuperadas, se mostrou de extrema importância.

A segunda etapa, de recuperação de informações estruturais através da análise do código fonte do software, caracterizou o sistema SASHE com cerca de dez mil linhas de código C++, implementando o gerenciador da hiperbase (71,5% do código), e cerca de quatro mil linhas de código Delphi, implementando a interface do sistema (28,5% do código). A interface entre o gerenciador da hiperbase e a interface do sistema se dá através de uma DLL, a qual exporta as funções de manipulação da hiperbase para a interface. Conforme especificado pelo método e descrito na Seção 2, foi elaborado um quadro de chamadas para cada arquivo de código fonte do sistema (Etapa2-a1). Todos os quadros definidos para os arquivos de código C++ totalizaram setecentos e vinte e dois (722) procedimentos registrados, incluindo protótipos e funções implementadas, sendo que para cento e setenta (170) procedimentos (23,5%) não foram encontradas chamadas que os ativassem. Esse alto número de procedimentos inativos deve-se ao fato de que alguns objetos implementam métodos que não são utilizados pelo sistema, no entanto, complementam o conceito representado pelo objeto. Também existem

procedimentos implementados em versões anteriores que, mesmo deixando de serem utilizados na versão analisada, não foram retirados do código pelos mantenedores.

No total, foram analisadas quarenta e sete (47) classes, quinhentas e vinte e nove (529) funções (descartando-se os protótipos), codificadas em oitenta e quatro (84) arquivos de código C++. A construção desse quadro demandou cerca de 1 mês e meio de trabalho apenas para os arquivos C++. Parte dessa tarefa foi auxiliada pelo uso da ferramenta *Understand for C/C++* [18], da qual foi utilizada uma cópia *demo,* disponível via Internet.

Quanto aos procedimentos em Delphi, foram analisados, no total, cento e sessenta e cinco (165) procedimentos, definidos em vinte e um (21) arquivos. Grande parte dos procedimentos analisados não apresentaram nenhuma chamada no código. Devido ao paradigma da linguagem, orientada a eventos, já era esperado que fossem encontradas poucas chamadas aos procedimentos implementados em Delphi. Cerca de dez dias foi o tempo despendido na análise desses arquivos.

Aproximadamente, cinco dias foram necessários para a construção do quadro de índice de procedimentos (Etapa2-a2), com os procedimentos em C++ e Delphi.

Finalmente, a elaboração do quadro de operações-procedimentos de implementação, embora ainda não esteja totalmente terminada (Etapa2-b), está estimada em aproximadamente três semanas a sua conclusão. O nível de detalhamento desse quadro pode ser ajustado de acordo com o propósito da engenharia reversa. No caso deste trabalho, a documentação recuperada será utilizada na re-engenharia do sistema para mudança do domínio de aplicação. Como o foco principal dessa re-engenharia se concentra na interface do sistema, o quadro de operações-procedimentos de implementação será detalhado até o nível das chamadas das operações exportadas pela DLL.

5. Validação do Fusion-RE/I

Como mencionado na seção anterior, no início do processo de engenharia reversa não existia quase nenhuma documentação de análise e projeto do sistema SASHE, sendo que as informações existentes estão em dois relatórios técnicos [6], [16]. Em [16] é mostrada uma pretensa arquitetura em camadas para o sistema SASHE (Figura 6-a), assim como uma breve descrição das funcionalidades inicialmente desejáveis para o sistema. Também nesse relatório é apresentada uma visão geral da máquina HIP (Figura 6-b), componente do sistema SASHE. Em [6] é feita uma apresentação da interface do sistema e de suas principais funcionalidades. No entanto, em nenhum momento é apresentada uma modelagem do sistema.

Dessa forma, neste estudo de caso, não é possível comparar a documentação recuperada com a documentação original do sistema, uma vez que a mesma não existe. No entanto, as informações recuperadas através da aplicação do método Fusion-RE/I mostraram-se consistentes com os relatórios existentes, embora a arquitetura proposta inicialmente para o sistema SASHE (Figura 6-a), não tenha tido todos os seus componentes implementados.

Como premissa do Fusion-RE/I, para que o engenheiro de software possa recuperar os modelos de análise de um sistema, é preciso antes reunir informações sobre o mesmo. A documentação existente, a linguagem de implementação, os dados obtidos por meio de entrevistas com os usuários e o domínio do sistema são fontes de informação valiosas [19]. Dentre esses, o domínio do sistema, por ser um aspecto bastante subjetivo, merece atenção diferenciada.

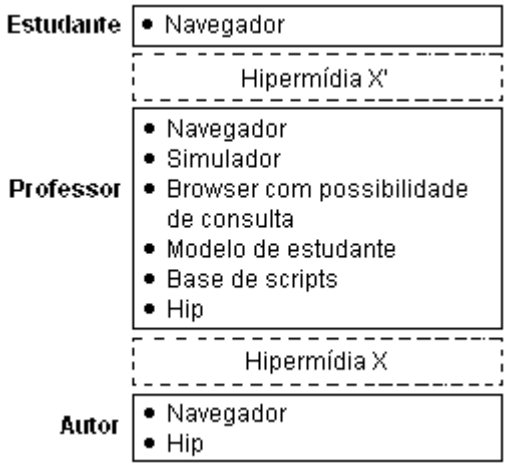

Figura 6-a. O ambiente SASHE proposto [16]

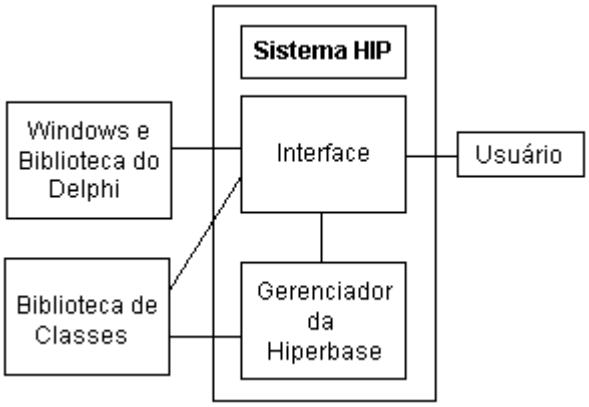

Figura 6-b. Arquitetura básica da máquina Hip [16]

Domínio do sistema pode ser entendido como a área de conhecimento ao qual um sistema pertence, que pode variar muito. Certamente, o domínio exerce influência no levantamento de informações sobre um determinado sistema. Uma vez que o engenheiro de software possui conhecimento (familiaridade) sobre o domínio ao qual o sistema pertence, maior é a chance desse sistema ser melhor entendido. E conseqüentemente, maior a chance de se obter uma melhor análise do mesmo.

Dessa mesma forma, a familiaridade com o domínio do sistema influi na construção de um modelo mental do sistema. O modelo mental permite ao engenheiro de software escolher um caminho particular de pensamento sobre um problema e gerar uma representação deste problema. Quanto mais familiar for o domínio ao engenheiro de software, uma melhor representação do sistema é feita e menos erros surgirão posteriormente [20].

Pesquisas já indicaram que a experiência e a familiaridade do engenheiro em um domínio de software influem diretamente nas etapas do ciclo de vida do software [20], [21]. Sob a perspectiva da engenharia reversa, DeBaud [22] realizou estudos de caso em que descreve como o conhecimento do domínio pode auxiliar um processo de engenharia reversa e como a engenharia reversa pode ser usada para construir um modelo melhor de domínio de uma aplicação.

Quando um sistema pertence a um determinado domínio, a sua interface pode ser apontada como o elemento que melhor evidencia este domínio. Há muito a interface deixou de ser apenas mais um elemento do sistema; é através dela que se dá o conjunto de processos, diálogos e ações entre o usuário e o computador, de forma a resultar em mecanismos objetivos para se obter a funcionalidade desejada.

No Fusion-RE/I, interação com a interface é ponto inicial do processo de engenharia reversa. Uma interação exaustiva com a interface é realizada, tornando o engenheiro de software conhecedor das funcionalidades e características operacionais do sistema. Dessa forma, o Fusion-RE/I possibilita um ganho efetivo de familiaridade com o domínio.

Embora a recuperação dos modelos de análise trate de uma tarefa ligada à manipulação do sistema, por meio da sua interface, de forma bastante intensiva, sendo dessa forma uma tarefa “concreta”, é exigido um grande esforço de atenção e concentração por parte do engenheiro de software (ou mantenedor), ao investigar quase que exaustivamente todos os estados da interface do sistema.

Os modelos de ciclo de vida e de operações, embora exijam um grande esforço para sua obtenção, fornecem um conhecimento completo da funcionalidade do sistema, revelando as

seqüências de comandos permitidas e a descrição das operações do sistema de forma clara e objetiva.

Um ponto crítico evidenciado durante a aplicação do método foi a recuperação do modelo de objetos (Etapa1-b3). Os estudos de caso anteriores realizados com o Fusion-RE/I [9], [23] foram aplicados a sistemas originalmente não orientados a objetos, e com um estilo de interface diferente do utilizado no SASHE. Informações sobre os estudos de caso realizados com o método são apresentados no Quadro 2. O fato do sistema SASHE ser orientado a objetos, nos permitiu analisar uma série de fatores quanto à Etapa1-b3.

Quadro 2. Informações sobre os estudos de caso realizados com o Fusion-RE/I

Sistema	Domínio	Paradigma de programação	Linguagem	Nº de linhas de código	Plataforma	Documentação Funcional Gerada		
						Nº de sentenças do ciclo de vida	Nº de operações do modelo de operações	Nº de modelos de objetos
SASHE	Autoria e suporte hipermídia ao ensino	Orientação a objetos	C++ e Delphi	14.000	PC/ Windows	55	62	2
SAPES [23]	Apoio a pesquisa científica	Orientado a função	Clipper	4.000	PC/ Dos	46	17	4
Proteum V1.1 C [9]	Teste de software	Orientado a função	C e X-View	27.000	Estações SUN/ Unix	Não disponível	Não disponível	5

O método Fusion-RE/I prescreve uma seqüência de ações para a recuperação do modelo de objetos - definição de temas e elaboração de um modelo de objetos para cada tema definido. O método dita que essa tarefa seja realizada durante a recuperação das visões funcionais, ou seja, sem que o código do software tenha sido investigado.

Uma primeira questão a analisar é a validade da realização dessa tarefa (Etapa1-b3) durante a primeira etapa do método (Recuperação de Visões Funcionais). Em sistemas cujo código não está sob o paradigma OO, faz sentido construir uma abstração do mesmo, na forma de modelos de objetos, através da análise de suas funcionalidades e do seu domínio de atuação, sem que o código implementado interfira na modelagem. Entretanto, quando o software está implementado sob o paradigma OO, a estrutura do modelo de objetos já existe e está instanciada no código do software. Assim, a construção de um modelo de objetos baseado somente nas visões funcionais do sistema deixa de ser válida. A recuperação do modelo de objetos, sem dúvida, é uma tarefa fundamental na documentação do software, no entanto, para sistemas orientados a objetos, a realização dessa tarefa nos parece mais adequada no início da segunda etapa do método (Recuperação de Visões Estruturais).

De fato, na experiência com o SASHE, ao se iniciar a segunda etapa, o modelo de objetos teve que ser ajustado às classes implementadas.

Outra questão é a validade da definição de temas para a recuperação dos modelos de objetos para sistemas OO. A partir do momento que o modelo de objetos deixa de ser uma abstração criada a partir das visões funcionais do software e passa a registrar a sua estrutura real, a definição de temas perde o significado. Nesse caso, pode-se discutir a utilidade dos temas também para sistemas OO, mas não da maneira como é proposto pelo Fusion-RE/I. Uma possibilidade de aplicação da definição de temas seria a sua utilização para a organização do modelo de objetos extraído do código. Sistemas maiores podem exigir que a estrutura recuperada do código seja dividida em vários modelos de objetos. No entanto, nesse

caso, os temas é que deveriam se adequar a implementação, já que o modelo implementado não pode ser mudado para se adequar aos temas propostos de acordo com as funcionalidades observadas. Sendo assim, seria mais adequada também a definição de temas na segunda etapa do método.

As tarefas para recuperação de visões estruturais, segunda etapa do método, foram adequadas, porém bastante trabalhosas, como pode ser verificado através dos números apresentados na Seção 4. Foi constatado que parte dessas tarefas podem ser automatizadas com sucesso.

De acordo com o Fusion-RE/I, a primeira tarefa de recuperação estrutural é a elaboração do Quadro de Chamadas (Etapa2-a1). Parte da elaboração desse quadro foi feita automaticamente, com o auxilio da ferramenta *Understand for C/C++* [18], conforme descrito na Seção 4. No entanto, a descrição dos procedimentos teve que ser feita manualmente, o que demandou muito trabalho para sua conclusão. A informação descrita nesse quadro é bastante significativa, pois através dele é possível rastrear a execução de cada procedimento, do momento em que é chamado (*Chamado por*) aos procedimentos que são disparados no corpo do procedimento (*Chama*). Além disso, o entendimento da funcionalidade de cada procedimento é apresentada no item *Descrição*.

Quanto ao Quadro de Índice (Etapa2-a2), a sua geração pode ser totalmente automatizada, já que esse quadro é uma lista de todos os procedimentos e suas respectivas localizações (arquivo/diretório). Neste trabalho, esse quadro se torna especialmente útil, pois será usado na construção de uma página índice da hiperbase contendo *links* para acesso aos procedimentos de outro quadro.

O Quadro de Operações-Procedimentos de Implementação (Etapa2-b) encerra a documentação estrutural recuperada segundo o método Fusion-RE/I. Esse quadro complementa as informações apresentadas no quadro de chamadas, associando a cada operação identificada na interface os procedimentos responsáveis por sua implementação. Um ponto positivo do método é a flexibilidade de se detalhar os procedimentos relacionados a cada operação num nível de profundidade que esteja de acordo com o objetivo final da engenharia reversa, como foi exemplificado na Seção 4.

Um ponto a se discutir na elaboração desse quadro (Etapa2-b) é a classificação dos procedimentos em um dos temas definidos na primeira etapa do método, ou como uma implementação da interface. Novamente entramos na questão da validade da elaboração de temas para sistemas orientados a objetos. No caso de sistemas implementados sob outros paradigmas, a classificação dos procedimentos nos temas definidos é importante para que se possa ter uma idéia de onde (em qual classe) cada procedimento ficaria melhor implementado. Isso seria especialmente útil no caso de uma re-engenharia para mudança de paradigma (não-OO para OO). Uma vez que o código é orientado a objetos, essa classificação deixa de ser necessária, pois cada procedimento já é implementado como método de uma determinada classe, permanecendo válida apenas a classificação dos procedimentos que implementam as interações da interface relativas a cada operação.

Na próxima seção são apresentadas as conclusões deste artigo, ressaltando os resultados obtidos com o desenvolvimento deste trabalho.

6. Conclusões

A engenharia reversa visa especificamente a construção de algum tipo de representação, em geral mais abstrata, obtida a partir de um sistema implementado. Neste trabalho, um

método de engenharia reversa, denominado Fusion-RE/I, foi utilizado com o objetivo de recuperar informações de projeto de um sistema hipermídia.

O método, descrito na Seção 2, foi cuidadosamente estudado e seu processo, modelado segundo OOHDM. Um aspecto relevante foi de que os resultados parciais (documentos recuperados) do processo de engenharia reversa foram então registrados no sistema hipermídia ao qual foi submetido à engenharia reversa. Como resultado deste experimento, além dos modelos abstratos desse sistema hipermídia terem sido recuperados, diversos aspectos de validação do método Fusion-RE/I foram evidenciados. Dentre eles, pode-se destacar:

- Efetivo suporte às tarefas de recuperação das visões funcionais e estruturais do sistema, com procedimentos bem definidos e objetivos;
- A adequada etapa inicial, de interação com a interface do sistema, cuja meta é fornecer conhecimentos sobre a funcionalidade e características operacionais do sistema, e que, além disso, proporciona uma efetiva familiaridade com o domínio, viabilizando um melhor entendimento para a elaboração dos modelos de análise;
- A questão do momento de se realizar a recuperação dos modelos de objetos que, em se tratando de um sistema implementado sob o paradigma de Orientação a Objetos (OO), pareceu ser mais adequado no início da recuperação da visão estrutural (Etapa2);
- Em se tratando de um sistema implementado sob o paradigma de Orientação a Objetos (OO), o modelo de objetos deixa de ser uma abstração criada a partir das visões funcionais do software e passa a registrar a sua estrutura real. Dessa forma, a definição de temas perde o significado. Uma nova utilidade dos temas para sistemas OO pode então ser visualizada: a sua definição para auxiliar a organização do modelo de objetos extraído do próprio código. Sistemas muito grandes podem exigir que a estrutura recuperada do código seja dividida em vários modelos de objetos.

Assim, os pontos de validação do método Fusion-RE/I, aplicado a um sistema OO, proporcionam uma adequação simples na ordem de realização de algumas tarefas. O Fusion-RE/I não havia sido proposto para ser aplicado em sistemas originalmente implementados sob o paradigma OO. Observa-se, dessa forma, a flexibilidade do método e sua conveniência para atender essa abordagem.

O trabalho descrito neste artigo deverá ser continuado com o objetivo de completar a re-engenharia do sistema SASHE, com vistas a adaptá-lo ao domínio de Engenharia de Software.

Referências Bibliográficas

[1] Saleh, K.; Boujarwah, A. Communications Software Reverse Engineering: A Semi-Automatic approach. *Information and Software Technology*, Oxford, n.38, p. 379-390, 1996.

[2] Schach, S.R. The Economic Impact of Reuse on Maintanance. *Journal Software Maintanance: Research and Practice*, v.6, n.4, p.185-96, 1994.

[3] Schneidewind, N.F. The State of Software Maitanance. *IEEE Trans. on Software Engineering*, v.13, n.3, p.303-10, 1987.

[4] Costa, R.; Iavanroni, G.M.; Sanches, R.; Maldonado, J.C., Masiero, P.C. Engenharia Reversa: da Interface do Software aos Modelos de Análise do Método Fusion. In: Simposio en Orientación a Objetos, *Anales*, Buenos Aires, Argentina, p.133-146, 1998.

[5] Meira, S.M.; Cabral, R.S. Id: Um Sistema de Hipertexto Configurável e Orientado a Objetos para Integrar Documentos de Software. In: VIII Simpósio Brasileiro de Engenharia de Software, *Anais*, Curitiba, pp. 283-96, 1994.

[6] Nunes, M.G.V.; Hasegawa, R.; Vieira, F.M.C.; Santos, G.H.R.; Fortes, R.P.M. SASHE: Sistema de Autoria e Suporte Hipermídia para Ensino. Notas, nº 33, ICMC–USP, São Carlos, 1997.

[7] Nunes, M.G.V.; Fortes, R.P.M.; Nicoletti, M.C. Flexible Guided-tours in Hypertexts: a way of controlling the user in learning applications. In: World Multiconference on Systemics, Cybernetics and Informatics SCI'97/ISAS'97, *Proceedings*, Caracas, Venezuela, 1997.

[8] Nunes, M.G.V.; Fortes, R.P.M. Roteiros em Aplicações no Ensino: a Questão do Controle do Leitor. In: III Workshop em Sistemas Multimídia e Hipermídia, *Anais*, São Carlos, p. 15-27, 1997.

[9] Costa, R.M. Um método de Engenharia Reversa para Auxiliar a Manutenção de Software. Dissertação de mestrado, ICMC–USP, São Carlos, 1997.

[10] Schwabe, D.; Rossi, G. The Object-Oriented Hypermedia Design Model. *Communications of the ACM*, v.38, n.8, p. 45-46, 1995.

[11] Schwabe, D.; Rossi, G.; Barbosa, S.D.J. Systematic Hypermedia Application Design with OOHDM. In: Hypertext'96, Washington DC, USA, March 1996. *Proceedings.* New York, ACM Press, p.116-28, 1996.

[12] Rossi, G. Um Método Orientado a Objetos para o Projeto de Aplicações Hipermídia. Tese de Doutorado. Departamento de Informática Pontifícia Universidade Católica, Rio de Janeiro, 1996.

[13] Coleman, D. et al. Desenvolvimento Orientado a Objetos: O Método Fusion. Ed. Campos, Rio de Janeiro, 1996.

[14] Penteado, R.A.D. Um método para Engenharia Reversa Orientado a Objetos. Tese de Doutorado, IFSC–USP, São Carlos, 1996.

[15] Feltrim, V.D.; Fortes, R.P.M. Requisitos de Hiperdocumentos de suporte ao domínio de Engenharia Reversa de Software. In: Workshop de Engenharia de Requisitos, *Anais*, Maringá-PR, p. 159-167, 1998.

[16] Nunes, M.G.V.; Hasegawa, R.; Vieira, F.M.C. Hip/Windows: Um Ambiente de Autoria de Hiperbases Multimídia, Relatório Técnico, nº 38, ICMC–USP, São Carlos, 1996.

[17] Rational Corporation. URL: http://www.rational.com, 1999.

[18] STI - Scientific Toolworks, Inc. URL *http://www.scitools.com*, 1999.

[19] Biggerstaff, T. Design Recovery for Maintenance and Reuse. *IEEE Computer*, v.22, n.7, pp. 36-49, 1989.

[20] Adelson, B.; Soloway, E. The Role of Domain Experience in Software Design. *IEEE Transactions on Software Engineering*, v.SE-11, n.11., p.1351-1360, 1985.

[21] Greenbaum, J.; Kyng M. (eds), Design at work: Cooperative of Computer Systems. Lawrence Erlbaum, Hillsdale, NJ, 1991.

[22] DeBaud, J-M.; Moopen, B.; Rugaber, S. Domain Analysis and Reverse Engineering. In: International Conference on Software Maintenance, Victoria. *Proceedings*, p. 326-335, 1994.

[23] Quinaia, M.A. Diretrizes para Reengenharia de Software com Características de Software Legado. Dissertação de Mestrado, ICMC–USP, São Carlos, 1998.