Esboço de um Processo Ágil de Desenvolvimento baseado em Framework

Franciene Duarte Gomes*
Maria Istela Cagnin
UNIVEM - Fundação de Ensino Eurípides Soares da Rocha,
Marília, São Paulo, Brasil, Caixa Postal 2041, CEP 17525-901
franciene@gmail.com, istela@fundanet.br

José Carlos Maldonado
ICMC/USP- Universidade de São Paulo – Campus São Carlos
São Carlos, São Paulo, Brasil, Caixa Postal 668, CEP 13560-970
jcmaldon@icmc.usp.br

Resumo

A utilização de processos de desenvolvimento para garantir a qualidade do software é evidente. Atualmente, existem diversas técnicas que, quando utilizadas nos processos de desenvolvimento, colaboram para isso. Dentre elas têm-se linguagens de padrões, *frameworks* e métodos ágeis, de interesse deste trabalho. Este artigo apresenta o esboço de um processo ágil de desenvolvimento baseado em *frameworks* denominado PARFAIT/EA. Esse processo é abstraído do PARFAIT[1], utilizado na migração de sistemas legados para o paradigma orientado a objetos. Algumas atividades do PARFAIT serão mantidas ou removidas, outras alteradas. Haverá a necessidade de adicionar outras que são específicas do desenvolvimento e, portanto, não encontradas no PARFAIT. Um estudo de caso conduzido com o PARFAIT será utilizado para a análise e definição do esboço do processo, o qual atenderá somente a engenharia avante.

Palavras chaves: Processo de Desenvolvimento, Métodos Ágeis, Linguagem de Padrões, *Frameworks.*

Abstract

The use of development processes to guarantee the software quality is evident. Currently, there are several techniques that, when used in the development processes, collaborate for that. Pattern languages, frameworks and agile methods, which are the focus of this work, appear among them. This paper presents the sketch of an agile development process based on frameworks called PARFAIT/EA. This process is abstracted from the PARFAIT, used in the migration of legacy systems for the object oriented paradigm. Some PARFAIT activities will be kept, others will be removed or modified. There will be a need to add specific activities of this development process and, therefore, are not found in PARFAIT. A case study lead with PARFAIT will be used for the process sketch analysis and definition, which will only take care of forward engineering.

Keywords: Development Process, Agile Methods, Pattern Language, Frameworks.

---

* Apoio Financeiro CAPES

[1] **P**rocesso **Á**gil de **R**eengenharia baseado em **FrA**meworks no domínio de sistemas de **I**nformação com técnicas de VV&**T**

1. Introdução

Com o aumento pelo interesse na qualidade do software, intensificou-se o uso de processos de desenvolvimento, sendo um dos principais fatores para o sucesso do software [12]. Um dos objetivos do uso de processo de desenvolvimento é aumentar a produtividade do software.

Neste mesmo contexto, vários métodos denominados métodos ágeis são encontrados na literatura [1], por exemplo: XP [3], Scrum [1], FDD [1], entre outros. Segundo Pressman [11], método é uma técnica utilizada no desenvolvimento de software, já o processo de desenvolvimento, abrange métodos, ferramentas e procedimentos.

Os métodos ágeis, com suas características principais, ou seja, interativos, incrementais e adaptativos, buscam aumentar a agilidade no processo de desenvolvimento de software. Além disso, várias técnicas de reúso vêm sendo propostas para apoiar o desenvolvimento de software. Dentre essas técnicas podem ser citados os *frameworks* [8] e as linguagens de padrões [2].

Segundo Appleton [2], *frameworks* é uma arquitetura reusável em que fornece estrutura e comportamento comuns para uma família de abstrações de *software*. Já, Fayad e Johnson [8] classificam *frameworks* em três grupos, com base em seu escopo: *framework*s de infra-estrutura do sistema, utilizados no desenvolvimento de sistemas portáveis, como sistemas operacionais, entre outros; *frameworks* de integração *middleware*, utilizados em ambientes distribuídos, com o objetivo de integrar aplicações e componentes distribuídos e *frameworks* de aplicação empresarial, utilizados para gerar aplicações com base no domínio do *framework* e que são de interesse deste trabalho para apoiar o desenvolvimento de software com menos tempo e com baixo custo.

O uso de *frameworks* de aplicação no desenvolvimento de software proporciona uma maior agilidade ao processo de desenvolvimento, permitindo disponibilizar versões do software em um tempo menor de desenvolvimento. Um outro ponto importante é com relação ao histórico de uso destas técnicas, que já foram testadas, o qual fornece uma melhor garantia da qualidade do *software*.

As linguagens de padrões são formadas por um conjunto de padrões de software para resolver um problema mais complexo [2]. Segundo Appleton [2], padrões são soluções para problemas recorrentes que ocorrem em um determinado domínio.

Nesta perspectiva é proposto um esboço de um processo ágil de desenvolvimento de *software* que utilize como apoio computacional um *framework* cuja construção foi baseada em uma linguagem de padrões de análise (LPA), facilitando tanto a análise do software desenvolvido, quanto a documentação, o entendimento e o uso do *framework*. A definição do esboço desse processo será abstraída a partir de um processo ágil de reengenharia baseado em framework, denominado PARFAIT. Para isso, os autores se basearam nos resultados de um estudo de caso de reengenharia conduzido anteriormente. O artigo está organizado da seguinte maneira: na Seção 2 apresentam-se os trabalhos relacionados, na Seção 3 apresenta-se o processo de reengenharia PARFAIT, na Seção 4 apresenta-se o esboço do processo ágil de desenvolvimento de software PARFAIT/EA e na Seção 5 discutem-se as conclusões.

2. Trabalhos Relacionados

Até a escrita deste artigo nenhum processo ágil de desenvolvimento baseado em *framework* foi encontrado na literatura. Alguns trabalhos envolvendo processo ágil, mas não atendendo especificamente ao desenvolvimento do software com a utilização de *framework,* são discutidos. Um dos processos ágeis é voltado para o desenvolvimento com o uso de componentes e o outro voltado para a reengenharia com a utilização de *framework*.

O processo ágil de desenvolvimento baseado em componentes [13] utilizou como base o padrão de Engenharia de *Software* ESA (Agência Espacial Européia), atendendo a todas as características dos métodos ágeis, em que os componentes são criados e integrados ao *software* de acordo com as exigências e mudanças que podem ocorrer ao longo do projeto.

Um outro processo ágil é o processo PARFAIT, utilizado na reengenharia de sistemas legados para o paradigma orientado a objetos. PARFAIT utiliza *framework* baseado em linguagem de padrões de análise, sendo possível obter a documentação de análise do sistema legado bem como o entendimento do domínio ao qual pertence, a partir da aplicação da linguagem de padrões. Um dos objetivos do PARFAIT é fornecer uma versão do software o mais rápido possível. Mais detalhes sobre esse processo é obtido na Seção 3.

Neste contexto, o uso do processo ágil de reengenharia PARFAIT serviu como base para a elaboração do esboço de um processo ágil para atender somente o desenvolvimento de software, e está apresentado neste trabalho. Um dos fatores que possibilita utilizar o processo PARFAIT está relacionado nas etapas existentes em ambos os processos, sendo: levantamento de requisitos, análise, projeto, implementação, teste e implantação. O que muda tanto no processo de reengenharia quanto no de desenvolvimento do *software* é somente a maneira como serão iniciadas e utilizadas as etapas ao longo do projeto. De acordo com Sommerville [12], a principal diferença entre a reengenharia e o desenvolvimento de um novo *software* é o ponto de partida. Ressalta-se que a reengenharia abrange tanto a engenharia reversa como a engenharia avante, assim foi possível utilizar o PARFAIT como base deste trabalho.

3. Processo Ágil de Reengenharia PARFAIT

O processo PARFAIT, utilizado na reengenharia de sistemas procedimentais para o paradigma orientado a objeto, usa vários recursos para apoiar a reengenharia, por exemplo, o *framework* GREN [4]. O *framework* é utilizado na construção do sistema alvo decorrente da reengenharia do sistema legado. O *framework* GREN é baseado em uma LPA denominada GRN [4], pertencente ao domínio de Gestão de Recursos de Negócios, ou seja, os padrões podem ser usados para apoiar a análise de sistema relacionados a locação, comercialização e manutenção de recursos; sendo que a palavra recurso refere-se a um bem ou serviço gerenciado pelo *software*, definindo todas as suas características importantes. A LPA GRN auxilia o engenheiro de software no entendimento do domínio do *framework*, na análise e na documentação do projeto de reengenharia.

Um outro ponto importante do processo PARFAIT é o uso de algumas práticas ágeis do método ágil XP [3]. Essas práticas ágeis são: versões pequenas, cliente presente, testes constantes, jogo do planejamento, programação em pares, propriedade coletiva do código, integração contínua, metáfora e semana de 40 horas.

Na Figura 1 apresenta-se o processo PARFAIT dividido em quatro fases, sendo: CONCEPÇÃO, ELABORAÇÃO, CONSTRUÇÃO e TRANSIÇÃO. A estrutura da documentação do processo PARFAIT, com o uso de atividades, diretrizes, inspeções, artefatos de entrada e de saída, entre outros foi baseado no arcabouço RUP (*Rational Unified Process*) [9]. Segundo Cagnin [5], apesar do PARFAIT utilizar a estrutura do RUP, o objetivo de cada fase é específico para o contexto de reengenharia.

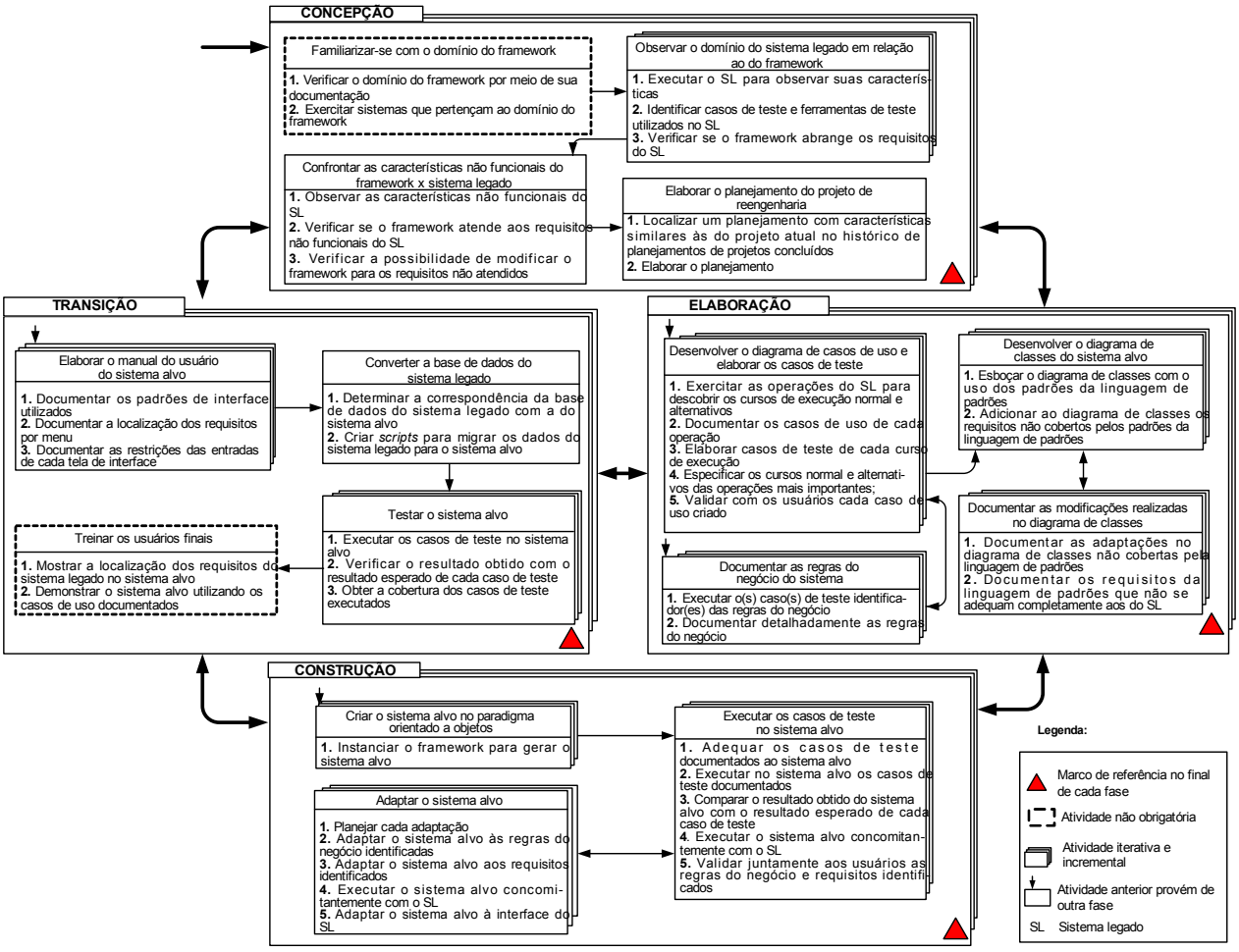

Figura 1 – Processo PARFAIT (adaptado de [5])

Todas as fases são formadas por atividades e cada atividade possui passos que são utilizados para executar a atividade por completo. No final de cada fase existe um marco de referência, que é representado por uma pirâmide localizada no interior do quadro de cada fase. O marco de referência é composto por questões com o objetivo de avaliar se a execução da fase obteve um resultado satisfatório. Em algumas fases, quando esse resultado não é atingido, há a necessidade de revisar todas as atividades executadas ou voltar para a fase anterior, como exemplo na fase de

CONCEPÇÃO, em que é feita uma avaliação para decidir em continuar ou cancelar o projeto de reengenharia. Para cada fase existem atividades, sendo que algumas são obrigatórias, como exemplo, na fase de CONSTRUÇÃO a atividade "Adaptar o sistema alvo*"* é obrigatória e deve ser executada pelo responsável do projeto. Outras atividades não são obrigatórias e estão representadas na figura por meio de retângulo com linhas tracejadas, por exemplo, a atividade "Familiarizar-se com o domínio do framework" da fase de CONCEPÇÃO.

Uma outra observação em relação as atividades do processo PARFAIT é que algumas atividades são iterativas e incrementais, ou seja, ao final da execução da atividade inicia-se a próxima atividade, retornando-se à atividade anterior após sua conclusão, a fim de refinar os artefatos produzidos caso seja necessário, até que todo o trabalho seja executado. Esse tipo de atividade é representado na figura por um retângulo de linha dupla. Um outro tipo de atividade é aquela que é (ou pode ser) a primeira a ser executada na fase do processo, a qual pertence. Essas atividades estão representadas na figura por uma seta localizada no interior do quadro que representa a fase do processo. Como exemplo, no quadro da fase de TRANSIÇÃO, no lado esquerdo no canto superior, a seta indica que a atividade "Elaborar o manual do usuário do sistema alvo" é a primeira atividade a ser executada nessa fase.

O processo PARFAIT possui algumas diretrizes e inspeções utilizadas em algumas atividades. As diretrizes são utilizadas para apoiar o uso de técnicas/tecnologias específicas durante a aplicação do processo e as inspeções são utilizadas para validar os artefatos produzidos. Para a definição do esboço do processo PARFAIT/EA apenas as fases e as atividades do PARFAIT serão consideradas.

A primeira fase do processo PARFAIT é a fase de CONCEPÇÃO. O objetivo desta fase é identificar o domínio do sistema legado em relação ao *framework* disponível. Nesta fase é realizada a elaboração do projeto de reengenharia, que geralmente é baseado em projetos anteriores ou na experiência do responsável pelo projeto. Esta fase é composta por quatro atividades, sendo: "Familiarizar-se com o domínio do *framework"*, "Observar o domínio do sistema legado em relação ao do *framework*", "Confrontar as características não funcionais do *framework* x sistema legado" e "Elaborar o planejamento do projeto de reengenharia".

A próxima fase é a de ELABORAÇÃO, cujo objetivo é elaborar a documentação do sistema legado, seguindo as prioridades dos requisitos definidas pelo usuário. Essa fase é composta pelas atividades: "Desenvolver o diagrama de casos de uso e elaborar os casos de teste", que executa o sistema legado com casos de teste funcionais, criados com o apoio dos critérios de teste Particionamento em Classes de Equivalência e Análise do Valor Limite [10], para entender a funcionalidade do legado e identificar os casos de uso; "Desenvolver o diagrama de classes do sistema alvo", com o apoio da LPA; "Documentar as modificações realizadas no diagrama de classes", que documenta os elementos do diagrama de classes que não pertencem à LPA; e "Documentar as regras do negócio do sistema".

A fase seguinte é a fase de CONSTRUÇÃO. O objetivo desta fase é realizar a instanciação do *framework* para obter o sistema alvo. Nesta fase são executados os casos de teste no sistema alvo, criados na fase de ELABORAÇÃO, e realizadas as adaptações nesse sistema para que fique com funcionalidades semelhantes ao do sistema legado. As atividades desta fase são: "Criar o sistema alvo no paradigma orientado a objetos", "Executar os casos de teste no sistema alvo" e "Adaptar o sistema alvo".

A última fase é a fase de TRANSIÇÃO e tem como objetivo garantir que o sistema alvo possa ser disponibilizado para o usuário. Nesta fase são realizados testes no sistema alvo, a fim de garantir sua integridade, realizar os ajustes finais, migrar a base de dados do sistema legado para a do novo sistema, etc. É composta pelas seguintes atividades:

“Elaborar o manual do usuário do sistema alvo”, “Converter a base de dados do sistema legado”, “Testar o sistema alvo” e “Treinar os usuários finais”.

O processo PARFAIT utiliza a “reengenharia guiada por teste”. Segundo Cagnin [5], essa característica foi baseada na prática de XP “Testes Constantes”. O objetivo é apoiar o entendimento do sistema legado, em que os testes são escritos logo no início da reengenharia, antes da migração do sistema legado, e posteriormente utilizados durante a fase de CONSTRUÇÃO para apoiar a adaptação do sistema alvo, resultante da instanciação do *framework*, a fim de ter funcionalidade semelhante àquela do legado.

É importante ressaltar que a “reengenharia guiada por teste” é um dos recursos pertencente ao Arcabouço de Reengenharia Ágil, denominado (ARA), assim como o processo PARFAIT. Tal arcabouço é utilizado para apoiar a migração de sistemas procedimentais para o paradigma orientado a objetos, atendendo a reengenharia e a engenharia reversa [7]. O ARA é formado por vários recursos, sendo: abordagens de reúso (linguagens de padrões, *frameworks*, etc), ferramentas de modelagem, ferramentas de controle de versões das aplicações criadas a partir da instanciação do framework, práticas ágeis, entre outros. O arcabouço baseia-se na reengenharia incremental e, segundo Cagnin [5], os requisitos submetidos à reengenharia são priorizados pelo cliente. Os clientes participam de todo o projeto, validando os artefatos conforme evoluem.

4. Esboço do Processo PARFAIT/EA

Esta seção apresenta o esboço do processo ágil de desenvolvimento baseado em *framework* denominado PARFAIT/EA, sendo que a abreviação EA significa Engenharia Avante. Tal processo será utilizado somente para apoiar o desenvolvimento do software e será abstraído a partir do processo PARFAIT. O processo PARFAIT/EA utilizará a estrutura do processo PARFAIT, sendo que algumas atividades serão aproveitadas, outras serão adaptadas, retiradas ou adicionadas para compor o novo processo.

Para uma melhor definição do esboço do processo de desenvolvimento, foi utilizado um estudo de caso de reengenharia, realizado anteriormente, com o apoio do processo PARFAIT. Esse estudo de caso foi conduzido em um sistema de biblioteca de uma universidade, implementado originalmente na linguagem de programação Clipper contendo aproximadamente 6 KLOC [6].

Ressalta-se que o conteúdo de algumas atividades do PARFAIT não foi alterado, no entanto apenas seus nomes foram modificados para se adequar ao contexto de desenvolvimento de software e estão citados nesta seção.

Para a definição das atividades da fase de CONCEPÇÃO do processo PARFAIT/EA, todas as atividades dessa fase do processo PARFAIT são utilizadas. A primeira atividade “Familiarizar-se com o domínio do framework” será utilizada na íntegra, pois o processo PARFAIT/EA também é baseado no uso de *framework* de aplicação. As outras atividades desta fase deverão ser adaptadas, como exemplo tem-se a atividade “Observar o domínio do projeto de software em relação ao framework”, com o nome já alterado para o contexto do processo PARFAIT/EA. Os dois primeiros passos dessa atividade do processo PARFAIT foram retirados, pois não atendem ao domínio de desenvolvimento. Esses passos são: “Executar o SL para observar suas características” e “Identificar casos de teste e ferramentas de teste utilizados no SL”. Para complementar esta atividade para o desenvolvimento de software, alguns

passos foram adicionados, sendo: “Realizar entrevistas com o cliente para identificar o domínio do projeto do software” e “Coletar documentos para apoiar na identificação do domínio do projeto de software”. Nessas atividades, o engenheiro de software deverá realizar entrevistas e coletar informações a respeito do domínio do sistema por meio de documentos (relatórios, manuais, procedimentos, etc) ou de entrevistas com o cliente.

Nas outras duas atividades da fase de CONCEPÇÃO do PARFAIT, com os nomes já alterados para o novo contexto, que são “Confrontar as características não funcionais do framework x projeto de software” e “Elaborar o planejamento do projeto de software”, foram adaptadas. A seqüência de execução dessas duas atividades foi alterada porque no caso do desenvolvimento do software o ideal é que na elaboração do planejamento do projeto de software as características não funcionais do projeto já estejam identificadas, a fim de serem confrontadas com as do *framework,* assim essas atividades devem ser executadas de forma paralela.

Ainda na fase de CONCEPÇÃO, na atividade já definida para o processo PARFAIT/EA “Elaborar o planejamento do projeto de software”, foram adicionados três novos passos, sendo: “Realizar entrevistas e fornecer questionário(s) para a coleta de requisitos”, “Definir a lista de funcionalidades do novo projeto junto ao cliente” e “Permitir que o cliente liste as funcionalidades mais importantes”. Com a adição desses novos passos, pode-se comparar esta atividade à prática ágil “Jogo do Planejamento” do método ágil XP [3], pois segue a mesma idéia, em que o cliente prioriza os requisitos para a próxima iteração e os programadores utilizam estimativas de custo para determinar o que precisa ser feito e o que pode ser adiado. Uma das técnicas que pode ser utilizada nessa atividade é o uso de questionário, que deve ser preenchido pelo próprio cliente ou por um dos integrantes do projeto. O questionário, depois de preenchido, será analisado pela equipe de projeto e documentado.

Na fase de ELABORAÇÃO, somente a primeira atividade “Desenvolver o diagrama de casos de uso e elaborar os casos de teste” foi adaptada para atender ao desenvolvimento do software. Nessa atividade o primeiro passo do processo PARFAIT é “Exercitar as operações do Sistema Legado para descobrir os cursos de execução normal e alternativo”. Este passo foi retirado, levando-se em conta que o sistema ainda não existe e que será desenvolvido. O segundo passo já definido no processo pelo nome “Documentar os casos de uso de cada funcionalidade” será aproveitado, em que todas as funcionalidades do software serão documentadas por meio de casos de uso. O outro passo alterado e definido para o PARFAIT/EA é “Elaborar casos de teste dos casos de uso priorizados pelo cliente”. Esse passo foi alterado, pois na fase de ELABORAÇÃO o projeto já possuirá uma lista das funcionalidades priorizadas pelo cliente identificadas na fase de CONCEPÇÃO. O último passo “Validar com os clientes cada caso de uso criado” será utilizado no PARFAIT/EA, pois é uma atividade importante uma vez que esse processo permite a participação do cliente em todo o projeto, principalmente para validar os artefatos produzidos. A participação do cliente em todo projeto baseia-se na prática ágil do método XP [3] “Cliente Presente”. Como o processo é adaptativo, nesse momento o usuário pode modificar as funcionalidades especificadas por meio dos casos de uso, caso haja necessidade.

Uma outra questão relevante analisada no estudo de caso está relacionada a atividade de teste. Cagnin et al. [6] observaram em um estudo de caso o grande esforço despendido nas atividades de teste durante a reengenharia utilizando o PARFAIT. Para o desenvolvimento do software optou-se também por utilizar os mesmos critérios de teste funcional no PARFAIT/EA, ou seja, Particionamento em Classes de Equivalência e Análise do Valor Limite. A criação dos testes é realizada durante a fase de ELABORAÇÃO e são utilizados nas fases de CONSTRUÇÃO e de TRANSIÇÃO para testar cada versão do sistema e para testar o sistema completamente antes de ser liberado para uso, respectivamente. A

criação de teste antes de construir o sistema assemelha-se a prática “Testes constantes” do método ágil XP [3], em que os testes são criados com base nos cartões de estórias escritos pelo cliente e todo o s*oftware* é validado por meio de teste.

Com base no estudo de caso de reengenharia, observa-se que as outras atividades da fase de ELABORAÇÃO, sendo: “Desenvolver o diagrama de classes do sistema”, “Documentar as modificações realizadas no diagrama de classes” e “Documentar as regras de negócio do sistema” são importantes para o desenvolvimento do software, pois a utilização dessas atividades favorecerá a produção da documentação do projeto do software, contribuindo para futuras manutenções.

Ressalta-se que a atividade “Desenvolver o diagrama de classes do sistema” é apoiada pela linguagem de padrões de análise e, como a construção do *framework* utilizado na fase de CONSTRUÇÃO é nela baseada, é possível obter uma versão do novo sistema, especificado com o apoio da linguagem de padrões, o mais rápido possível a partir da instanciação do *framework*. É necessário documentar as funcionalidades e as regras de negócio do sistema que não pertencem à linguagem de padrões e, conseqüentemente, ao *framework*, nas atividades “Documentar as modificações realizadas no diagrama de classes” e “Documentar as regras de negócio do sistema” a fim de que seja possível posteriormente adaptar a versão do sistema gerada a partir do *framework*. Isso é feito visando obter uma versão final do sistema de acordo com os requisitos definidos pelo cliente.

A fase de CONSTRUÇÃO do PARFAIT/EA é composta pelas atividades da fase de CONSTRUÇÃO do PARFAIT, com poucas adaptações. Na atividade “Executar os casos de teste no software”, o passo 4 “Executar o sistema alvo concomitantemente com o SL” será retirado, pois não atende ao desenvolvimento do software e sim à reengenharia. Na atividade “Adaptar o software”, os dois últimos passos foram adaptados. No processo PARFAIT/EA esses passos serão apoiados pelo cliente, sendo renomeados para: “Executar o software junto ao cliente” e “Adaptar o software de acordo com as mudanças exigidas pelo cliente”. Esses passos são importantes, pois é nesta atividade que são implementadas as regras de negócio já documentas na fase anterior. O uso desta atividade pode ser associada a prática ágil do método XP “Programação em Pares”, permitindo facilitar a implementação das adaptações, evitando erros ocorridos na escrita do código fonte. Torna-se possível ainda nesta atividade a aplicação de outra prática ágil do XP, sendo a prática “Propriedade Coletiva de Código”, em que todos os integrantes do projeto terão acesso ao código fonte, podendo também adicionar trechos de código no software, quando necessário.

Ainda na fase de CONSTRUÇÃO, o estudo caso de reengenharia apresenta versões do sistema alvo para serem confrontadas com as funcionalidades do sistema legado. Para apoiar o desenvolvimento do software, torna-se interessante a liberação de protótipos funcionais, em que esses protótipos serão validados pela equipe de projeto junto ao cliente e liberados para teste de aceitação. A liberação de protótipos funcionais pode ser comparada com a prática ágil “Versões Pequenas” do método ágil XP [3], em que versões freqüentes do software são liberadas. Para complementar esta fase, a prática ágil do método XP “Integração Contínua” deverá ser aplicada, pois permite a integração do software cada vez que uma tarefa é finalizada, evitando perdas de funcionalidades na liberação da versão.

No processo de reengenharia PARFAIT, a fase de TRANSIÇÃO é composta por quatro atividades, sendo uma delas “Converter a base de dados do sistema legado”. Em uma análise realizada no estudo de caso de reengenharia, definiu-se que essa atividade ficará isolada no processo PARFAIT/EA, pois ela só será utilizada quando realmente houver necessidade. A razão disso é a possibilidade de ocorrer casos em que exista um software operando na empresa cliente e seja necessário a conversão da base de dados do software antigo para a do novo software desenvolvido.

Uma outra adaptação necessária para o desenvolvimento foi realizada na atividade “Treinar os usuários finais”. Essa atividade no processo PARFAIT é definida como não obrigatória, pois depende de recursos financeiros cedidos pela empresa cliente. Analisando o estudo de caso e adaptando para o desenvolvimento, essa atividade passou a ser obrigatória, pois o cliente não possui nenhum software com essas funcionalidades instalado, ao contrário da reengenharia em que o cliente já tem familiaridade com o sistema legado. O primeiro passo dessa atividade, que é “Mostrar a localização dos requisitos do sistema legado no sistema alvo” será retirado, pois enquadra na mesma questão em que o software ainda não existe.

Uma outra adaptação necessária para esta última fase é a ordem na seqüência para executar as atividades. Com o isolamento da atividade já definida no processo PARFAIT/EA “Converter a base de dados”, a seqüência ficou da seguinte maneira: “Testar o software”,“Treinar os usuários finais” e por último “Elaborar o manual do usuário do software”. No projeto de reengenharia, com o uso do processo PARFAIT, uma versão funcional para uso é liberada somente no final permitindo a entrega do manual do usuário. Já no desenvolvimento de software, o processo PARFAIT/EA permitirá que versões funcionais sejam liberadas com freqüência, possibilitando a escrita do manual do usuário no final de cada ciclo de desenvolvimento ou somente na versão final do software.

Observando os resultados do estudo de caso de reengenharia, verificou-se a vantagem do uso de marcos de referência em todas as fases do processo PARFAIT/EA, o que possibilita ajustes necessários no planejamento do projeto. Com isso, definiu-se o uso de marco de referência em cada fase do PARFAIT/EA, sendo também adicionada uma outra atividade denominada “Reuniões freqüentes”, e que também será executada em todas as fases do processo PARFAIT/EA. Esta atividade está representada na Figura 2 por um círculo preto, podendo ser realizada no início ou no fim da fase ou na conclusão de cada atividade.

A classificação das atividades do processo PARFAIT, como atividade: não obrigatória, iterativa e incremental e atividade anterior provém de outra fase, serão mantidas no processo PARFAIT/EA. Com a análise do estudo de caso, observou que essa classificação é importante pois orienta o engenheiro de software no uso das atividades, forçando de uma certa maneira executar todos os passos necessários para o desenvolvimento do software.

Na Figura 2 apresentam-se todas as atividades alteradas (fonte maiúscula), as não alteradas (fundo pontilhado) e as retiradas (fonte sublinhada) do processo PARFAIT para abstrair o processo de desenvolvimento PARFAIT/EA. Além disso, houve a necessidade de adicionar algumas atividades intrínsecas ao contexto de desenvolvimento ágil de software e estão apresentadas na Figura 2 com marcadores.

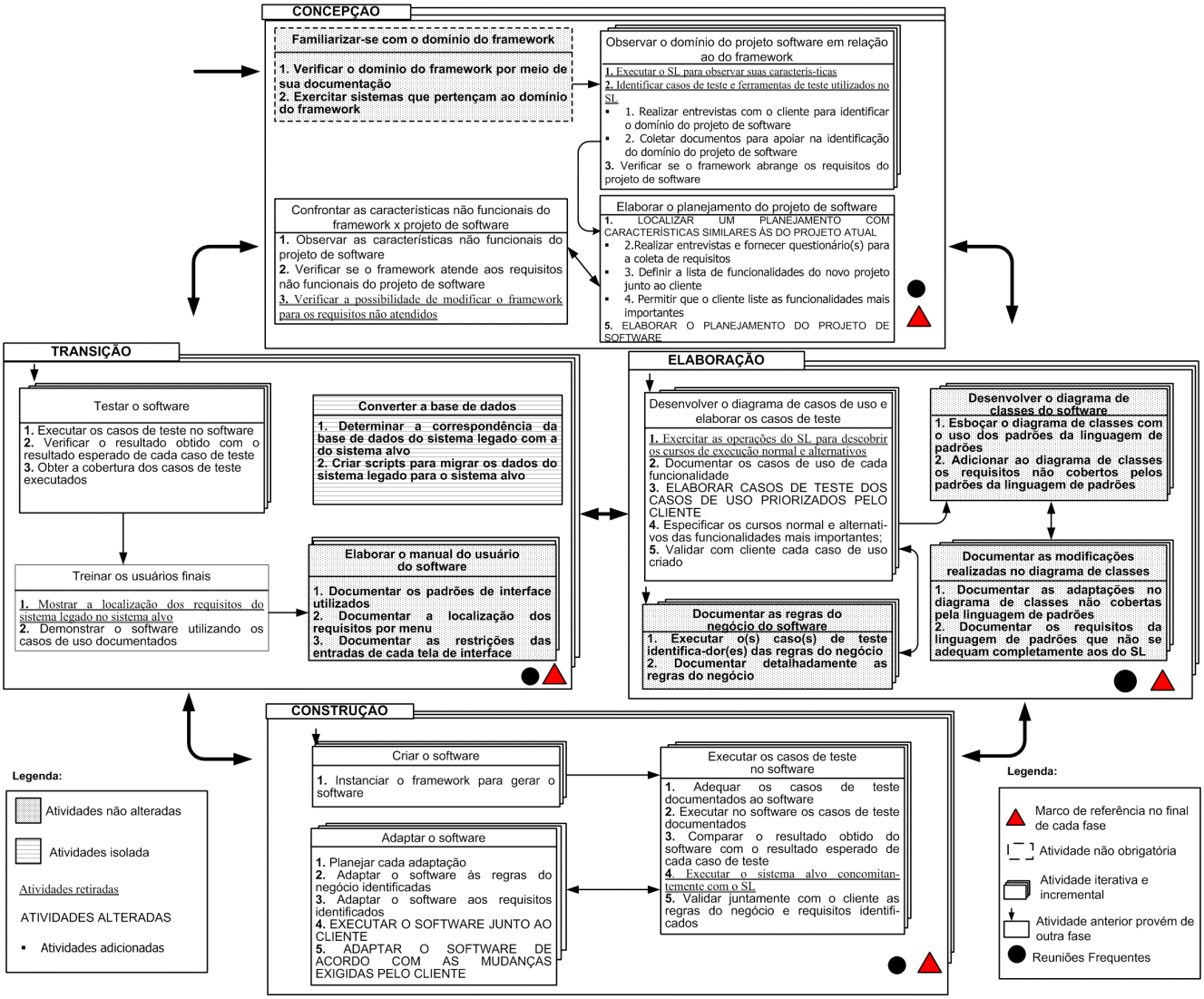

Figura 2 – Alterações no PARFAIT para obtenção do PARFAIT/EA

A partir da abstração feita, observando os resultados da execução de cada uma das atividades do PARFAIT, foi possível obter um esboço das atividades e passos do processo ágil de desenvolvimento baseado em *framework* PARFAIT/EA, como apresentado na Figura 3.

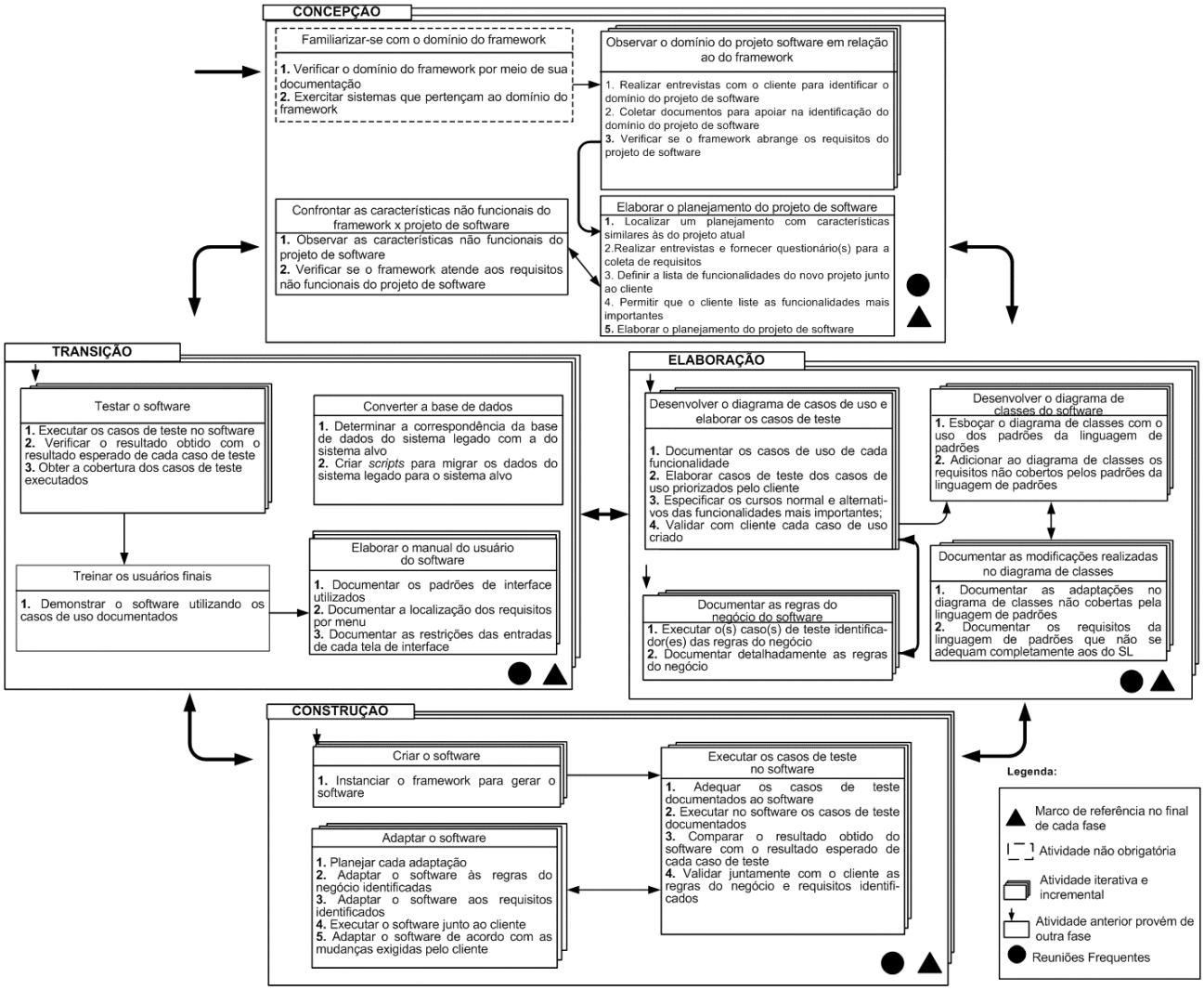

Figura 3 – Esboço do processo PARFAIT/EA

5. Conclusões

Com a análise e a definição do esboço do processo PARFAIT/EA, observou-se que muitas das atividades do processo PARFAIT puderam ser reutilizadas, pelo fato de ambos os processos atenderem a fase de engenharia avante, de alguma forma, bem como utilizar como apoio computacional *framework* baseado em linguagem de padrões de análise.

Com a definição do PARFAIT/EA, observou-se a possibilidade do arcabouço ARA, que apóia tanto a reengenharia como a engenharia reversa, em apoiar também o desenvolvimento ágil de software, atendendo somente a engenharia avante. Desta forma, o processo PARFAIT/EA poderá ter acesso a vários recursos do ARA que atendam ao desenvolvimento do software (por exemplo, ferramentas de migração de dados, ferramentas de controle de versão das aplicações criadas a partir de *frameworks*, etc).

Para a evolução, consolidação e maturidade do processo PARFAIT/EA diversos estudos de casos de desenvolvimento de software deverão ser conduzidos, com o objetivo de avaliar o uso do processo, verificando se realmente as atividades definidas atendem ao desenvolvimento do software e observando a possibilidade de adicionar

outras atividades e artefatos para apoiar a documentação das informações envolvidas no desenvolvimento e, posteriormente, na manutenção dos sistemas desenvolvidos, aprimorando o processo.

Notou-se que a agilidade do processo PARFAIT/EA não será definida somente pelo uso das práticas ágeis, mas também pela utilização do *framework* de aplicação, que apóia a construção do software.

Com base nisso, estuda-se a possibilidade de definir outros recursos computacionais para apoiar o processo, como exemplo, o uso de ferramentas para controlar e gerenciar a aplicação de processos, a fim de construir uma base de dados histórica dos projetos concluídos para apoiar futuros projetos quanto a atividade de planejamento de projeto.

Referências

[1] ABRAHAMSSON, P.; SALO, O.; RONKAINEN, J.; WARSTA, J. **Agile Software Development Methods**: **Review and Analysis.** ESPOO (Technical Research Centre of Finland), 2002.

[2] APPLETON, B. **Patterns and software: Essential concepts and terminology.** URL: http://www.cmcrossroads.com/bradapp/docs/patterns-intro.html, 1997.

[3] BECK, K. ***Extreme Programming explained: Embrace change***. Second ed. Addison-Wesley, 2000.

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

[5] CAGNIN, M. I. **PARFAIT: Uma Contribuição para Reengenharia de Software baseada em Linguagens de Padrões e Frameworks.** Tese de Doutoramento, ICMC-USP, São Carlos-SP, 2005.

[6] CAGNIN, M. I.; MALDONADO, J. C.; GERMANO, F.S.R.; CHAN, A. **Um Estudo de Caso de Reengenharia Utilizando o Processo PARFAIT.** In: SDMS'S2003, Simpósio de Desenvolvimento e Manutenção de Software da Marinha, Rio de Janeiro, RJ, CD-ROM, 10 p., 2003a.

[7]CAGNIN, M. I.; MALDONADO, J. C.; GERMANO, F. S.; MASIERO, P. C.; PENTEADO, R. D.; CHAN, **An agile reverse engineering process based on a framework**. In: WER'2003, 6th International Workshop on Requirements Engineering, Piracicaba, SP, p. 240-254, 2003b.

[8] FAYAD, M. E.; JOHNSON, R. E. **Domain-specific Application Frameworks: Frameworks Experience by Industry**. First ed. John Wiley & Sons, 2000.

[9] KRUCHTEN, P. **The Rational Unified Process: An Introduction**. Second ed. Addison-Wesley, 2000.

[10] MYERS, G. J. **The art of software testing**. second ed. Wiley, 2004.

[11] PRESSMAN, R.S. **Engenharia de Software**. 5ª ed. Rio de Janeiro: McGraw-Hill, 2002.

[12] SOMMERVILLE, I. **Engenharia de Software.** Editora: Addison Wesley,2003.

[13] ZWARTJES, G; GEFFEN, J.V. **An Agile Approach Supported by a Tool Environment for the Development of Software Components.** Master's Thesis, Technische Universiteit Eindhoven,University of Pretoria, South Africa, 2005. URL: http://espresso.cs.up.ac.za/publications/gzwartjes_jvgeffen_dissertation.pdf**.** Acesso em Março/2006, 2005.