Requisitos para utilização de prototipagem evolutiva nos processos de desenvolvimento de software baseado na Web

Bruno C. Soares[1]

[1]Departamento de Ciência da Computação – Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte – MG – Brasil

brunocesarino@hotmail.com

Abstract. *This paper presents a requirements catalog for the use of evolutionary prototyping in the Web based software development processes that is based on a comparative study of the literature. The purpose of this catalog is to help users compare, select or and customize evolutionary prototyping methodologies applied to Web based software development.*

Resumo. *Este artigo apresenta um catálogo de requisitos para a utilização de prototipagem evolutiva nos processos de desenvolvimento de software baseado na Web. Os requisitos são baseados em um estudo comparativo da literatura. O propósito deste catálogo é auxiliar os usuários a comparar, selecionar e personalizar as metodologias de prototipagem evolutiva aplicadas ao desenvolvimento de software baseado na Web.*

1. Introdução

Segundo Paula (2001), a prototipagem, nos processos de desenvolvimento de software, pode ser entendida como uma técnica aplicável a atividades do fluxo de requisitos. O protótipo é um modelo operacional do software a ser desenvolvido (Budde e Zullighoven, 1990). Seu principal objetivo é auxiliar a especificação e validação de requisitos relevantes ou problemas de implementação, permitindo elaborar e testar interfaces com os usuários de maneira visual e interativa.

Protótipos são, normalmente, desenvolvidos de forma rápida e representam uma versão simplificada do software, que implementa certos aspectos e funcionalidade. O uso de prototipagem pode trazer inúmeras vantagens aos processos de desenvolvimento de software, desde a redução dos riscos relacionados às mudanças de requisitos a definições de projetos de interface. No entanto o uso de prototipagem também pode acarretar alguns inconvenientes, como a omissão da complexidade de algumas regras de negócio e a frustração dos usuários devido a eventuais mudanças na interface do software final. Conforme discutido em diversos estudos, por exemplo, Budde e Zullighoven (1990) e Lichter *et al.* (1993), a prototipagem pode ser feita por meio de diversas técnicas e metodologias e se aplica a diferentes tipos de desenvolvimento de software. Uma classificação bastante usual das técnicas de prototipagem consiste em separá-las em descartáveis e evolutivas ou evolucionárias (Paula, 2001).

Este artigo trata da prototipagem evolutiva aplicada aos processos de desenvolvimento de software baseado na Web. Na Seção 2 são formalizados os

conceitos relativos à prototipagem de software, assim como algumas de suas principais classificações, relativas aos tipos de protótipos, sua utilização e metodologias de desenvolvimento. Os requisitos levantados são elencados e apresentados na forma de um catálogo na Seção 3. Para a escolha desses requisitos foram abordados diversos aspectos, como o planejamento e a dimensão do projeto, as características da equipe de desenvolvimento, as ferramentas e linguagens utilizadas, etc. Na Seção 4 são discutidos alguns aspectos práticos e operacionais, relacionados aos relatos de alguns projetos, obtidos em algumas referências. A conclusão se encontra na Seção 5.

2. Prototipagem de Software

Além de auxiliar a especificação de requisitos, o uso de protótipos pode servir de base para subsidiar tomadas de decisão e como forma de ganhar experiência prática (Budde e Zullighoven, 1990).

Protótipos podem ter alta ou baixa fidelidade. Quanto mais os protótipos se assemelham ao software real, mais alta é considerada sua fidelidade, aspecto discutido detalhadamente em Walker *et al.* (2002).

Lichter *et al.* (1993) e Budde e Zullighoven (1990) classificam os protótipos nos seguintes tipos:

- Protótipo de apresentação;
- Protótipo próprio (*prototype proper*);
- Protótipo tábua de pão (*breadboards*); e
- Sistema piloto.

A prototipagem separa-se ainda pelas metas/objetivos (*goals*) (Lichter *et al.*, 1993) ou alvos/finalidades (*aims*) (Budde e Zullighoven, 1990):

- Prototipagem exploratória;
- Prototipagem experimental; e
- Prototipagem evolutiva ou evolucionária (*evolutionary*).

Lichter *et al.* (1993) citam ainda uma classificação pelas técnicas de construção:

- Horizontal: implementação superficial de apenas uma camada do software, geralmente a camada de interface com o usuário; e
- Vertical: uma parte específica do software é implementada por completo.

Essas diversas classificações podem se sobrepor na prática. Os tipos são basicamente distintos pelas características do protótipo como produto e os objetivos ou finalidades se relacionam aos processos de desenvolvimento.

Conforme Paula (2001) e Gordon e Bieman (1995), as metodologias de prototipagem são classificadas em descartável e evolutiva ou evolucionária. No presente trabalho, vamos nos ater a esta última classificação, conforme definido nas subseções seguintes, e com enfoque na prototipagem evolutiva.

2.1. Prototipagem Descartável

Protótipos descartáveis (*throw-away*), após sua utilização, são deixados de lado ou passam a servir apenas para consultas. Neste tipo de prototipagem o processo de desenvolvimento do software real é totalmente independente do processo utilizado para a elaboração do protótipo. Normalmente possuem baixa fidelidade, pois são criados para auxiliar a especificação de requisitos e não deveriam integrar o desenho do software.

2.2. Prototipagem Evolutiva

Protótipos evolutivos ou evolucionários (*evolutionary*) são criados nas fases iniciais do projeto e refinados ao longo do decorrer do processo de desenvolvimento do software, podendo ser interpretados como liberações. Incrementos de funcionalidade são incorporados ao protótipo, que, tendo sua fidelidade gradualmente aumentada, se torna o software final. Assim, os processos de desenvolvimento do protótipo e do software real são essencialmente o mesmo.

2.3. Prototipagem de Software baseado na Web

As peculiaridades nos processos de desenvolvimento de software baseado na Web levam à necessidade de adaptações nas técnicas de prototipagem tradicionais para sua adequada utilização. Segundo Jeenickle *et al.* (2003), uma das características inerentes aos processos de desenvolvimento de software baseado na Web é o momento tardio em que os usuários conseguem expressar plenamente suas necessidades. Como um entendimento bem fundamentado das necessidades dos usuários é um fator chave nos processos de desenvolvimento de software baseado na Web, especialmente quando se trata de sistemas governamentais, onde os usuários podem ser qualquer cidadão, a especificação de requisitos tem um papel fundamental no processo e pode ser eficientemente auxiliada pela utilização da prototipagem.

3. Catálogo de Requisitos

Esta seção apresenta um catálogo de requisitos para o uso de prototipagem evolutiva nos processos de desenvolvimento de software para a Web. As partes interessadas, de uma forma geral, são: de um lado os usuários finais e patrocinadores; de outro os gerentes de projeto e os desenvolvedores (prototipadores). Alguns requisitos, entretanto, são mais direcionados a um ou mais grupos específicos de interessados, como tentou se explicitar a cada descrição de requisito presente no catálogo.

O catálogo é organizado de forma a elencar prioritariamente requisitos que se aplicam à prototipagem evolutiva nos processos de desenvolvimento de software para a Web. Alguns requisitos, no entanto, não são específicos do desenvolvimento de software para a Web, se aplicando a qualquer processo que utilize a prototipagem evolutiva. Outros são ainda mais genéricos e se aplicam à prototipagem de uma forma geral.

A elaboração do catálogo foi baseada em diversos trabalhos e estudos relacionados ao uso da prototipagem nos processos de desenvolvimento de software. Alguns desses trabalhos são comentados a seguir.

No estudo realizado por Lichter *et al.* (1993), cinco projetos industriais de desenvolvimento de software que utilizaram a prototipagem evolutiva foram analisados e comparados. É interessante ressaltar que a maioria dos projetos analisados não foi bem sucedida. Os autores contrastam a teoria e a prática da prototipagem em projetos de software. Esse estudo possibilitou a revelação de diversos requisitos para o uso de protótipos evolutivos, trazendo, inclusive, alguns aspectos práticos.

Gordon e Bieman (1995) apresentam uma pesquisa sobre o uso de prototipagem em processos de desenvolvimento de software. Nessa pesquisa, analisaram-se 39 estudos de caso envolvendo projetos de desenvolvimento de software, onde se utilizou a prototipagem descartável ou a evolutiva. Os autores enunciam e discutem uma série de atributos da prototipagem, fundamentados em relatórios e depoimentos dos envolvidos nos projetos analisados. Esses atributos contribuíram significativamente para a elicitação de alguns dos requisitos aqui apresentados.

Uma analise dos diversos usos da prototipagem (usos estendidos), além da tradicional avaliação da usabilidade do software apresentada por Buskirk e Moroney (2003), também foi de grande utilidade para a obtenção de requisitos para a utilização da prototipagem de uma forma geral.

Jeenicke *et al.* (2003) sugerem uma técnica de prototipagem evolutiva, *e-Prototyping*, criada especialmente para auxiliar a revelação de requisitos nos processos de desenvolvimento de software baseado na Web. Os autores relatam alguns dos fatores de sucesso, o que contribuiu para a obtenção de requisitos relativos ao uso da prototipagem no desenvolvimento de software baseado na Web.

Esses e outros trabalhos foram de grande importância para a escolha dos requisitos que compõem o catálogo apresentado a seguir. Algumas citações e referências se encontram na própria descrição dos requisitos, servindo de embasamento e fundamentação para sua escolha.

3.1. Envolvimento dos Usuários

- O processo de prototipagem deve prever um canal de comunicação entre usuários, desenvolvedores e demais partes interessadas.
    - O processo de prototipagem deve garantir a comunicação participativa entre desenvolvedores e usuários.
    - O acesso dos usuários finais ao protótipo deve ser limitado e gerenciado.
    - A comunicação deve priorizar a interação e garantir a cooperação entre desenvolvedores e usuários.
    - No caso da prototipagem evolutiva, a comunicação deve ser assegurada e monitorada em todo o processo de desenvolvimento, e não apenas na especificação de requisitos e na implantação, quando ela é indispensável.
    - A comunicação deve ser gerenciada de forma centralizada e com ênfase no retorno das requisições dos usuários. Este requisito se aplica primordialmente a sistemas baseados na Web, onde é comum a existência de usuários desconhecidos.

Fundamentação: Problemas como desentendimentos e baixa produtividade podem se relacionar às falhas na comunicação entre membros das equipes de desenvolvimento ou entre estes e os usuários ou seus representantes. Quando a comunicação falha, o protótipo ou sistema resultante pode representar mal as necessidades dos usuários. Outro fator relevante é que tanto usuários quanto desenvolvedores devem estar interessados em manter o elo de comunicação entre si. A interação e cooperação entre usuários e desenvolvedores asseguram a avaliação contínua dos protótipos e permitem que se mantenha o controle do progresso do desenvolvimento. No entanto, quando é dado aos usuários muito acesso ao protótipo, problemas relacionados à expectativa podem surgir e expectativas irreais podem levar ao fracasso da prototipagem ou até mesmo do projeto. Segundo Gordon e Bieman (1995), a falta de participação dos usuários pode ser desfavorável para o processo de desenvolvimento de software, anulando o benefício obtido com a utilização da prototipagem rápida. Lichter *et al.* (1993) definem a prototipagem evolutiva como um processo onde os desenvolvedores deixam de ser protagonistas e passam a atuar como consultores técnicos, trabalhando continuamente em cooperação com os usuários para aprimorar o sistema. Em um dos projetos analisados por Lichter *et al.* (1993), o sucesso do uso da prototipagem evolutiva deveu-se parcialmente à interação dos usuários com os desenvolvedores. Os usuários tinham contato com os protótipos, que eram entregues em períodos de três a quatro meses, mantendo assim o controle sobre o progresso do desenvolvimento enquanto avaliavam os protótipos. Em outros dois projetos, um dos motivos do fracasso foi exatamente a falta de envolvimento e manutenção de um elo de comunicação com o usuário. Budde e Zullighoven (1990) consideram que a comunicação (organizacional e técnica) entre desenvolvedores e usuários, que não pode ser evitada na definição dos requisitos e na implantação do sistema, deve ser mantida durante todo o processo de desenvolvimento para este tenha uma adequada abordagem evolutiva. Jeenicke *et al.* (2003) analisam o levantamento de requisitos a partir de usuários de sistemas baseados na Web e defendem que a comunicação entre desenvolvedores e usuários é essencial para direcionar o processo de prototipagem. É comum que os usuários de sistemas baseados na Web (ou sua maioria) sejam desconhecidos, o que dificulta a sua interação com os desenvolvedores. Segundo Jeenicke *et al.* (2003), esse problema pode ser amenizado com a centralização do gerenciamento e coleta dos retornos e requisições dos usuários e com a ênfase em respostas rápidas e no uso de rastreadores de erro (*bug-tracking*).

- O processo deve prever mecanismos para direcionar a escolha de representantes dos usuários e demais partes interessadas para participarem da prototipagem.
    - Deve-se garantir o envolvimento dos usuários finais ou reais do software no processo de prototipagem.
    - A escolha dos atores para participarem do processo de prototipagem deve levar em consideração as múltiplas partes interessadas e perspectivas envolvidas, visando a poupar recursos e buscar inovação.

Fundamentação: Percebe-se a importância da participação dos usuários finais do software no trabalho de Gordon e Bieman (1995). Segundo os autores, o envolvimento dos usuários finais, não só os gerentes intermediários ou patrocinadores, deve ser garantido no processo de prototipagem. O papel fundamental do envolvimento dos usuários também é destacado no trabalho de por Lichter *et al.* (1993). Os autores

comentam que em todos os projetos analisados o envolvimento de usuários reais foi de grande importância, e que a falta de participação desses também pode levar a sistemas não otimizados. Segundo Jeenicke *et al.* (2003), no desenvolvimento de software baseado na Web, especialmente quando se trata de sistemas governamentais, o acesso aos usuários finais fica prejudicado, principalmente pela heterogeneidade destes. Portanto, nesse tipo de desenvolvimento de software, as partes interessadas e seus requisitos precisam ser identificados cuidadosamente. Os autores também esclarecem que o levantamento de requisitos, assim como o uso da prototipagem, envolve a existência de muitos atores relevantes, e suas perspectivas deveriam ser incluídas no processo para poupar recursos e buscar a inovação.

- O processo deve garantir que as expectativas dos usuários sejam mantidas e controladas.
    - A aparência do protótipo deve levar em consideração o propósito da prototipagem de forma a não gerar falsas expectativas aos usuários.

Fundamentação: Dependendo do propósito da prototipagem, manter a aparência do protótipo mais simples pode ser importante para garantir a expectativa dos usuários. Segundo Buskirk e Moroney (2003), um protótipo que seja muito "polido" tende a desencorajar os avaliadores a dar o retorno necessário com relação à solução apresentada. Assim os comentários tendem a ser menos específicos e em menor número.

3.2. Clareza/Completude dos Requisitos

- O processo deve prever mecanismos que garantam que os requisitos mais claros ou bem definidos, ou pelo menos parte deles, sejam implementados primeiro na prototipagem evolutiva.

Fundamentação: Requisitos claros e bem definidos facilitam a criação de protótipos., Quando determinada necessidade dos usuários ou funcionalidade do sistema é totalmente conhecida, o protótipos é gerados se torna mais robusto e consistente, favorecendo, principalmente, a prototipagem evolutiva. Os usuários também ficam mais à vontade para explicitar seus desejos e revelar mais requisitos, ou clarear aqueles que ainda não estão bem especificados, quando interagem com moldes ou exemplos mais funcional. Segundo Gordon e Bieman (1995), o processo de elaboração do desenho do software fica mais rápido e a prototipagem evolutiva também é favorecida quando os requisitos levantados são bem claros e conhecidos, levando a uma redução do esforço de especificação de requisitos e de desenvolvimento. Em dois dos projetos analisados por Lichter *et al.* (1993), onde se utilizou a prototipagem evolutiva, a falta de clareza na especificação dos requisitos, devida principalmente à não capacidade expressão dos usuários, levou ao abandono do projeto. Jeenicke *et al.* (2003) defendem que, no processo de prototipagem evolutiva, os requisitos que estão completamente revelados devem ser implementados primeiro. Segundo Ginige e Murugesan (2001) *apud* Jeenicke *et al.* (2003), um entendimento bem fundamentado das necessidades dos usuários é um fator chave para o sucesso do desenvolvimento de sistemas baseados na Web. A própria intenção dos protótipos, no entanto, é ajudar a clarear ou definir melhor os requisitos do software a ser desenvolvido. Esse fato se evidencia mais na prototipagem descartável, onde os requisitos não precisam estar tão claros inicialmente. Os próprios usuários preferem explicitar seus requisitos com demonstrações práticas ou invés de lidar com

especificações na forma de documentos ou planilhas. Segundo relatos de projetos que utilizaram protótipos, sob a luz do trabalho de Gordon e Bieman (1995), os usuários muitas vezes não gostam de documentações escritas, e estas são ainda sujeitas a interpretações, ao contrário de protótipos, que por sua vez são mais definitivos.

3.3. Hardware

- O processo de desenvolvimento do software não deve permitir que definições de hardware sejam tomadas muito cedo, *i. e.*, antes da prototipagem.
    - Todas as partes interessadas no processo de desenvolvimento devem ser consultadas quando das definições de hardware do sistema.
    - Definições de hardware devem prever a integração com o protótipo evolutivo, caso este esteja sendo desenvolvido.

Fundamentação: As decisões de hardware podem afetar os protótipos, principalmente os evolutivos, e devem ser tomadas em consonância com o desenvolvimento do software. A integração do protótipo ou do software final com um hardware diferente do utilizado no ambiente de prototipagem ou desenvolvimento pode chegar a inviabilizar o projeto. Entre os motivos do abandono de um dos projetos analisados em Lichter *et al.* (1993) reportam-se as dificuldades de integração da aplicação com o hardware estipulado pelo cliente.

3.4. Planejamento

- O planejamento do projeto de desenvolvimento de software deve ser integrado com processo de prototipagem, principalmente se esta for evolutiva.

Fundamentação: O processo de prototipagem deve ser previsto no planejamento do projeto do software. Dois projetos analisados por Lichter *et al.* (1993), onde se utilizou a prototipagem, apontaram a falta de planejamento como um dos principais problemas encontrados no desenvolvimento. Um deles foi devido a um planejamento insuficiente e o outro à escolha inadequada dos marcos do projeto. A avaliação do protótipo, por outro lado, deve fornecer subsídios para o planejamento do projeto, que inclui o próprio uso futuro do protótipo. Segundo Jeenicke *et al.* (2003), após a avaliação do protótipo, deve-se definir se ele será utilizado apenas como um veículo de aprendizagem (abordagem descartável) ou aproveitado como parte do sistema alvo (abordagem evolutiva).

- O processo de prototipagem deve ser planejado e monitorado.
    - O objetivo do protótipo (para que prototipar) deve ser bem definido e seu cumprimento monitorado.
    - O escopo do protótipo (o que prototipar) deve ser bem delimitado e sua ampliação ou redução deve repercutir no planejamento.
    - O propósito do protótipo (por que prototipar) deve ser bem estabelecido e cumprido de forma integrada com o planejamento do projeto.
    - A manutenção do protótipo e do sistema deve ser prevista no planejamento da prototipagem evolutiva.

Fundamentação: Os objetivos, o propósito e o escopo devem ser observados para a escolha adequada do tipo e metodologia de prototipagem. A definição adequada desses parâmetros reduz os riscos de sub ou superestimativas do projeto e permite um controle mais efetivo do processo de prototipagem. Segundo Gordon e Bieman (1995), superestimativas podem ser controladas com uma adequada definição do propósito do protótipo e com o auxílio de treinamentos. Os autores comentam ainda que a manutenibilidade do sistema pode ser prejudicada pela utilização da prototipagem evolutiva.

- O processo deve garantir que a prototipagem com o propósito de avaliação da interface com o usuário faça parte da especificação de requisitos.

Fundamentação: Não é aconselhável que a prototipagem que tenha como propósito avaliar a interface com o usuário sirva de base para o desenvolvimento do software. Como na prototipagem evolutiva os protótipos são incrementados até se tornarem o sistema final, deve-se tomar cuidado para que o fator determinante da estrutura do sistema não seja apenas a interface com usuário. Esse tipo de utilização da prototipagem tem bastante utilidade, e é muito comum na abordagem descartável, mas deve fazer parte da especificação de requisitos. Segundo Gordon e Bieman (1995), desenhar o sistema inteiro a partir da interface com o usuário pode ser perigoso, pois esta nem sempre caracteriza a melhor estrutura do sistema.

- O processo deve prever que aspectos críticos do sistema sejam prototipados.
    - Aspectos críticos do sistema podem ser implementados em conjunto ou não com a interface com o usuário

Fundamentação: Diversos aspectos do software ou do sistema a ser desenvolvido podem ser prototipados, além da tradicional modelagem da interface com o usuário. O processo de desenvolvimento deve então assegurar que aspectos críticos sejam verificados e, quando couber, prototipados. Segundo a experiência de Lichter *et al.* (1993), a mistura dos diversos tipos de protótipos (de apresentação, *prototype proper*, *breadboards* e sistemas piloto) pode ser necessária para o sucesso do desenvolvimento do sistema. A prototipagem, em um dos projetos analisados por esses autores, não teve como intenção modelar a interface com o usuário.

- O processo deve procurar estabelecer uma metodologia ou cultura de prototipagem na organização.

Fundamentação: O estabelecimento de uma metodologia ou cultura de prototipagem pode ser favorável para o processo de construção do protótipo, uma vez que os desenvolvedores e gerentes se tornam habituados ao processo. Segundo Gordon e Bieman (1995), a falta de metodologia organizada pode ser a causa de desperdício de esforço. Schrage (1996) defende que as organizações podem aprender a conhecer e mudar sua cultura de prototipagem, e utilizá-la em favor de diversas outras atividades, como guiar as especificações de requisitos e gerenciar riscos.

- O processo deve garantir que seja feita uma estimativa de custos adequada para a prototipagem.

Fundamentação: Erros nas estimativas podem levar à sub ou supercontratação do projeto e/ou do protótipo. Em dois projetos que fracassaram, analisados por Lichter et

al. (1993), ocorreram subestimativas do custo e do esforço necessários para se avaliar os protótipos. Uma alternativa para os problemas relacionados às estimativas de custos da prototipagem é contratar separadamente o protótipo é tratá-lo como uma prova de conceito. Segundo Gordon e Bieman (1995), muitas companhias contratam a prototipagem como uma fase separada, o que pode ser apropriado para diversas situações.

- O processo deve prever que mecanismos para auxiliar a escolha da utilização da e do tipo de prototipagem.
    - O momento de criação e o estágio do processo de desenvolvimento do software devem ser levados em consideração na escolha do tipo de prototipagem.
    - O tamanho do projeto deve ser levado em consideração na escolha da utilização e do tipo de prototipagem.

Fundamentação: O propósito da prototipagem, o andamento do desenvolvimento do projeto e outros fatores de planejamento devem ser observados na escolha do momento de criação dos protótipos. Segundo Buskirk e Moroney (2003), a abordagem selecionada para a prototipagem depende do estágio do desenvolvimento do software, de restrições de cronograma e do grau de funcionalidade requerida para se avaliar a usabilidade da solução proposta. Os autores dizem ainda que, se os protótipos de demonstração de funcionalidade forem criados antes do final da análise, eles podem não incluir requisitos funcionais importantes. Já em Budde e Zullighoven (1990), comenta-se que a experimentação e a avaliação dos protótipos executáveis devem ser feitas o mais cedo possível. O tamanho do projeto é outro fator determinante na escolha da utilização da prototipagem. A seleção do tipo de prototipagem deve considerar o tamanho do projeto. Em projetos de grande porte os problemas gerados pela existência de requisitos de desempenho tendem a ser ampliados, principalmente quando se utiliza a prototipagem evolutiva. No entanto, o estudo realizado por Gordon e Bieman (1995) mostra que a utilização de prototipagem evolutiva em projetos de tamanho substancial foi adequada. Por outro lado, segundo as análises dos estudos de caso realizados pelos mesmos autores, em projetos pequenos o uso da prototipagem descartável se mostrou economicamente inviável. Hekmatpour (1987) conclui que a prototipagem evolutiva pode ser aplicada a projetos de tamanho substancial (20.000 linhas de código, sendo um terço em linguagem C e dois em Franz Lisp).

- O processo de prototipagem evolutiva requer ciclos de vida curtos e em grande quantidade.

Fundamentação: Nos processos de desenvolvimento de software que utilizam prototipagem evolutiva, deve-se procurar definir um maior número de iterações e com ciclos de vida curtos. Ciclos curtos ajudam a revelar os requisitos dos usuários, evitando uma grande quantidade de erros nas entregas de versões do software. Licher *et al.* (1993) sugerem que um dos fatores que se mostra como obstáculo para a prototipagem é o controle do processo dado por ciclos de vida tradicionais. Segundo Scharge (1996), quanto mais protótipos e ciclos de prototipagem por unidade de tempo, mais tecnicamente polido se torna o produto final. Jeenicke *et al.* (2003) apresentam uma forma de prototipagem (*e-Prototyping*) onde o suporte para o levantamento de requisitos

de usuários de sistemas baseados na Web é auxiliado por uma abordagem evolutiva baseada em ciclos de desenvolvimento curtos. Segundo os autores, o gerenciamento dos projetos de desenvolvimento de software que utilizam prototipagem deve buscar entregas curtas, comunicação e inovação. Ciclos curtos também são parte das tendências das metodologias ágeis de desenvolvimento de software, como o *Extreme Programming* (Beck, K., 2000 *apud* Jeenicke *et al.*, 2003). Dagnino (2002) comenta sobre as adaptações necessárias para compatibilizar a prototipagem evolutiva com as metodologias ágeis de desenvolvimento de software ao apresentar uma técnica de prototipagem, *Agile Development in Evolutionary Prototyping Technique* (ADEPT). Nessa técnica, uma característica essencial é a existência de ciclos de vida curtos. O autor defende que entregas curtas trazem vantagens como a captura de fatias do mercado e a melhoria da condição competitiva e dos retornos dos usuários finais.

- O processo deve procurar formar equipes de prototipagem com caráter multidisciplinar.

Fundamentação: O processo de prototipagem envolve desenvolvedores, preferencialmente bem experientes e treinados em linguagens de alto nível e voltadas para confecção de interfaces amigáveis, analistas de negócio e de requisitos, desenhistas ou projetistas (*designers*) e arquitetos de software, testadores, gerentes de projeto e representantes dos usuários e dos patrocinadores. As equipes de prototipagem, portanto, devem ter caráter essencialmente multidisciplinar. Segundo Houde e Hill (1997) *apud* Buskirk e Moroney (2003), as equipes envolvidas na prototipagem devem ser multidisciplinares, onde pessoas com habilidades diversas colaboram no processo.

- O processo deve procurar evitar que protótipos descartáveis se convertam no software final.

Fundamentação: O não descarte de um protótipo que deveria ser descartável pode levar a um software pouco robusto, com um desenho de baixa qualidade e com a manutenção prejudicada. O fato de a inicial proposta de uma prototipagem descartável se tornar evolutiva para economizar recursos é um problema muito comum, conforme observado por Gordon e Bieman (1995). Para se evitar o problema do não-descarte de protótipos construídos para esse propósito, os gerentes devem se comprometer com o que foi definido no escopo e propósito do protótipo.

3.5. Desempenho

- O processo deve-se prever mecanismos para avaliar a necessidade de se verificar questões de desempenho nos protótipos.

Fundamentação: Quando o propósito do protótipo é avaliar alternativas de desenho, é importante que problemas de desempenho sejam verificados. Segundo Gordon e Bieman (1995), a utilização de prototipagem pode, às vezes, melhorar o desempenho do sistema. No entanto, quando este não é adequadamente medido, problemas de integração podem surgir. Segundo Buskirk e Moroney (2003), protótipos que incluem elementos não funcionais em substituição aos componentes que ainda serão implementados podem dar a impressão de um desempenho mais acurado e melhor do que ocorrerá na realidade. Segundo alguns autores, problemas de desempenho tendem a se ampliar quando se utiliza a prototipagem evolutiva.

3.6. Desenho (*Design*)

- O processo deve garantir que decisões de desenho sejam tomadas conjuntamente com as de prototipagem.
    - A qualidade do desenho deve ser mantida nos protótipos, principalmente nos evolutivos.
    - Deve-se procurar utilizar padrões/mecanismos de desenho (*design patterns*) para garantir a qualidade do desenho na prototipagem evolutiva.
    - O processo pode prever mecanismos como o uso de listas de conferência de itens de desenho para garantir a qualidade do desenho do software e do protótipo, assim como sua manutenção.

Fundamentação: Segundo Buskirk e Moroney (2003), questões de desenho muitas vezes ditam o tipo de prototipagem necessária. Na prototipagem evolutiva, a qualidade do desenho precisa manter o padrão de qualidade desejado para o desenvolvimento do software. Lichter *et al.* (1993) defendem ainda que a qualidade de todos os protótipos, que não os com a finalidade exclusiva de demonstração, não deve ser negligenciada durante o desenho. A qualidade do desenho também deve ser mantida nos processos de desenvolvimento de software para a Web. A metodologia de prototipagem apresentada por Jeenicke *et al.* (2003), *e-Prototyping*, busca garantir o uso de altos padrões de qualidade no desenvolvimento do software, o que leva a uma ênfase adicional na qualidade do desenho técnico. Uma das formas de se garantir a qualidade do desenho é a utilização de padrões ou mecanismo de desenho (*design patterns*). Gordon e Bieman (1995) discutem o uso de padrões de desenho e ferramentas de prototipagem e concluem que, na prototipagem evolutiva, a qualidade do software pode sofrer se eles não forem adotados. Segundo Budde e Zullighoven (1990), a imposição de pré e pós-condições são mecanismos que ajudam a assegurar a qualidade do protótipo, e estão presentes em certas linguagens de programação, como *Eiffel*. Listas de conferência de itens de desenho do software também auxiliam na garantia da qualidade. Conforme discutido em Gordon e Bieman (1995), o uso de listas de conferência pode evitar problemas relacionados à falta de qualidade nos protótipos, como a permanência de código sem funcionalidade ou que deveria ter sido descartado, o que pode ocorrer mesmo quando se usam boas ferramentas de prototipagem. Segundo os autores, a inclusão de critérios de documentação na lista de conferência garante uma completa documentação do protótipo, o que auxilia sua manutenção.

- O processo deve prever a modularização dos protótipos.

Fundamentação: A construção e a manutenção de protótipos modularizados fica bastante facilitada, na medida em que os componentes são mais atômicos, independentes e compreensíveis. Problemas de restrições em protótipos demonstração de funcionalidade, que só apareceriam no ambiente real de produção do software, podem ser evitados com a utilização de ambiente de desenvolvimento aberto e modularizado. A integração de rotinas mais rápidas, caso seja necessário, também é favorecida pela modularização dos protótipos. Segundo Gordon e Bieman (1995), o sucesso da prototipagem evolutiva dependa da existência de alto grau de modularização. Para Budde e Zullighoven (1990), na abordagem evolutiva de desenvolvimento, o

software deve ser construído em peças pequenas e compreensíveis, e os rumos do projeto serão assim ajustados passo a passo. A modularização também é importante para a avaliação de opções de desenho, conforme comentado por Buskirk e Moroney (2003). O rearranjo de componentes modularizados, por exemplo, permite a rápida elaboração de diversas opções de leiaute de páginas da Web.

- O processo deve garantir que a arquitetura do desenho do protótipo seja bem estruturada, flexível e compatível com a da aplicação final.

Fundamentação: Na prototipagem evolutiva, se a arquitetura do protótipo não for bem estruturada e compatível com a do software final, em favor de um desenvolvimento rápido, o software resultante se torna incompreensível e de difícil manutenção e utilização. Para Lichter *et al.* (1993), a arquitetura dos protótipos deve ser compatível com a arquitetura fundamental da aplicação final, independentemente do propósito do protótipo. A garantia de um desenho bem estruturado, assim como a ausência de código obsoleto e a realização de revisões freqüentes, se encontram entre os fatores de sucesso apresentados por Hekmatpour (1987). Ainda segundo esse autor, a opção mais efetiva para se manter a flexibilidade em protótipos evolutivos é a utilização de arquitetura aberta.

- O processo deve garantir que código obsoleto ou não utilizado seja descartado.

Fundamentação: Códigos obsoletos ou não utilizados tornam o software ineficiente e inteligível. Assim, se esses códigos não forem descartados pode-se prejudicar o desempenho e comprometer a arquitetura do desenho do software. Hekmatpour (1987) coloca como parte do sucesso da utilização da prototipagem evolutiva o comprometimento em não se manter partes do desenho já deterioradas pela evolução do protótipo, o pode ser conseguido com reestruturações da arquitetura do sistema.

- O processo de prototipagem deve prever o uso metodologias ou tecnologias orientadas a objetos.

Fundamentação: O uso de metodologias ou tecnologias orientadas a objetos pode popar tempo e esforço, tanto no desenho quanto na implementação do protótipo. Em Gordon e Bieman (1995), alguns dos casos atribuíram o sucesso da prototipagem ao uso da abordagem de orientação a objetos. Lichter *et al.* (1993) citam que a utilização da orientação a objetos e de linguagens como Ada podem servir como soluções alternativas ao uso de métodos de prototipagem não muito bem definidos. Segundo Budde e Zullighoven (1990), as tecnologias orientadas a objetos também trazem mecanismos, como a separação entre interação e funcionalidade e a predefinição de tipos interativos, que podem auxiliar a construção de protótipos interativos. Em ambientes de prototipagem onde se utilizam os recursos oferecidos pela orientação a objetos, como o apresentado por Ozaki *et al.* (2003), a geração de protótipos pode ser feita em diferentes níveis de abstração, de forma extremamente ágil e modularizada.

3.7. Experiência/Treinamento/Expertise

- O processo deve prever treinamentos para os desenvolvedores e gerentes e usuários.
    - Os desenvolvedores do protótipo devem ser experientes, bem treinados e, preferencialmente, especializados.

- Os usuários que avaliarão o protótipo devem ser experientes e conhecedores do domínio da aplicação a ser desenvolvida e da metodologia de prototipagem a ser utilizada.
- Os gerentes de projeto devem estar familiarizados e treinados na metodologia de prototipagem a ser utilizada.

Fundamentação: O treinamento dos desenvolvedores nas metodologias de prototipagem deve ser previsto no processo, pois os protótipos precisam ser gerados modificados de forma rápida e ágil. Segundo o depoimento de uma das fontes de Gordon e Bieman (1995), os pré-requisitos para o sucesso da prototipagem incluem usuários e desenvolvedores bem experientes. De uma forma geral, as evidências analisadas pelos autores mostram que pode ser perigoso colocar desenvolvedores inexperientes no ambiente de prototipagem, sobretudo se decisões de desenho de alto nível estiverem envolvidas. Os gerentes de projeto também precisam estar familiarizados com a metodologia de prototipagem a ser utilizada, para que o planejamento da prototipagem e do projeto de desenvolvimento do software não sofra distorções de escopo, custo e prazo. Para Gordon e Bieman (1995), o treinamento dos gerentes pode evitar problemas como a subestimação e subseqüente contratação do projeto por um preço muito baixo. Os autores comentam ainda que com o auxílio de treinamentos e de uma adequada definição do propósito do protótipo, as superestimativas também podem ser controladas.

3.8. Ferramentas de Prototipagem

- O processo deve prever a utilização de ferramentas de prototipagem.
  - As ferramentas de prototipagem utilizadas devem trazer benefícios, como a redução do esforço e do custo do desenvolvimento.
  - Os problemas de dependência de tecnologia envolvidos na escolha da ferramenta de prototipagem devem ser minimizados.
  - A ferramenta de prototipagem deve ser compatível com a linguagem de programação do software final.

Fundamentação: A utilização de ferramentas de prototipagem pode reduzir o esforço e o custo, o que é de extrema importância no desenvolvimento evolutivo, onde são criadas várias versões do protótipo. Ozaki *et al.* (2003) ressaltam que, para que se tenha um software de qualidade, pode ser necessária a criação de diversos protótipos com vistas a permitir a escolha do desenho final do software. Os autores apresentam um ambiente para a prototipagem evolutiva, baseada na interpretação abstrata para programas em Java que inclui ferramentas que reduzem o esforço e custo da construção de protótipos. As ferramentas de prototipagem também podem ajudar a preencher diversos outros requisitos, como a manutenção de um elo de comunicação como os usuários e a flexibilização do desenho. Conforme mencionado por Budde e Zullighoven (1990), o uso de ferramentas de prototipagem pode servir para auxiliar a comunicação técnica entre desenvolvedores e usuários. A ferramenta apresentada por Luqi e Ketabchi (1988), *Computer-Aided Prototyping System* – CAPS, além de contribuir na comunicação com os usuários, permite a flexibilidade do desenho do protótipo, a integração com ferramentas de teste e monitoração de riscos. Luqi *et al*. (2000) mostram uma utilização especifica da ferramenta CAPS voltada para a prototipagem evolutiva. O trabalho de

Zhang *et al.* (2004) apresenta uma metaferramenta para a geração de linguagens de prototipagem (*Visual Executable Prototyping Languages* – VEPL). Em comparação com a ferramenta CAPS (Luqi e Ketabchi, 1988), a metaferramenta possui a vantagem de utilizar métodos formais e formalismos de grafos. No entanto, a utilização de ferramentas específicas de prototipagem pode deixar a manutenção do protótipo dependente da continuidade da disponibilidade dessas ferramentas. Segundo Zhang *et al.* (2004), ferramentas de prototipagem existentes ou são muito especializadas e não atendem a um largo espectro de aplicações, ou são muito difíceis de se utilizar por profissionais não especializados. Ferramentas de prototipagem podem demandar a utilização de linguagens específicas. A ferramenta CAPS (Luqi e Ketabchi, 1988) utiliza a linguagem *Prototype System Description Language* (PDSL), uma linguagem própria para a especificação de protótipos. As ferramentas que utilizam linguagens específicas devem ser utilizadas com cautela, pois se não existir uma geração adequada de código na linguagem alvo, a sua utilização também pode prejudicar a manutenção do protótipo.

3.9. Linguagens de Prototipagem/Programação

- O processo deve prever mecanismos para orientar a escolha de linguagens de programação ou de prototipagem.
    - A prototipagem deve procurar utilizar linguagens de programação de alto nível quando o propósito é a avaliação de interfaces com o usuário.
    - Deve-se procurar utilizar a mesma linguagem de programação, ou linguagens facilmente conversíveis, no protótipo e no software final.

Fundamentação: Na prototipagem descartável, o desenvolvimento rápido e sem um forte controle da qualidade é adequado. Segundo Buskirk e Moroney (2003), a prototipagem é mais adequada para desenvolvedores com habilidades em linguagens de alto nível e em programação rápida e suja (*quick-and-dirty*). Na abordagem evolutiva, no entanto, o processo de desenvolvimento deve manter os padrões de qualidade exigidos para o software final. Nesse caso, deve-se procurar utilizar a mesma linguagem de programação, ou uma linguagem facilmente conversível para a que será utilizada no software final, de forma a evitar que os desenvolvedores tenham que lidar com a linguagem do protótipo, a linguagem do software final e com a interface entre elas. Segundo Gordon e Bieman (1995), conversão do protótipo, devido à utilização de linguagens diferentes, pode consumir muito tempo e esforço, principalmente se as linguagens possuem características bastante distintas, por exemplo, apenas uma delas possui mecanismos de herança.

3.10. Domínio da Aplicação

- O processo deve garantir que seja realizada uma análise do domínio da aplicação ou do tipo de software a ser desenvolvido para definir ou selecionar a metodologia de prototipagem.
    - O domínio da aplicação pode restringir o uso da prototipagem ou definir a metodologia a ser utilizada.
    - Usuários especialistas no domínio da aplicação devem participar do projeto de prototipagem.

Fundamentação: Uma análise do domínio da aplicação é um pré-requisito importante para a realização da prototipagem. Segundo Budde e Zullighoven (1990), processamentos *batch* são resistentes à prototipagem, especialmente se as repetidas modificações do software são muito pesadas e caras. Lichter *et al.* (1993) concluem que os prototipadores devem ter um conhecimento suficiente da área da aplicação a ser desenvolvida e do ambiente de desenvolvimento.

3.11. Revisões

- O processo deve prever a realização de revisões freqüentes nos protótipos.

Fundamentação: A realização de revisões nos ajuda a manter o padrão de qualidade requerido na prototipagem evolutiva. Revisões freqüentes também garantem a rápida identificação de problemas aumentam a manutenibilidade do protótipo. Segundo o estudo apresentado por Hekmatpour (1987), entre os requisitos para o sucesso do uso da prototipagem, estão a garantia de um desenho bem estruturado, a ausência de código obsoleto e a realização de revisões freqüentes. O autor ressalta ainda a importância da realização de revisões para evitar que os problemas de tomada de decisão, aspectos críticos e freqüentes na utilização da prototipagem evolutiva, não sejam ignorados.

3.12. Monitoração/Avaliação de Riscos

- O processo deve prever que a prototipagem se integre com mecanismos de monitoração e avaliação de riscos.
    - A prototipagem evolutiva deve procurar fornecer indicadores e métricas para auxiliar a monitoração e avaliação de riscos.

Fundamentação: A prototipagem pode fornecer importantes subsídios para o gerenciamento de riscos. Na abordagem evolutiva, indicadores e métricas podem ser extraídos a partir da evolução dos protótipos e utilizados para monitorar e avaliar os riscos do projeto. No trabalho de Schrage (1996), comenta-se que as organizações podem utilizar a prototipagem para gerenciar riscos. Luqui e Nogueira (2000) apresentaram um modelo de avaliação de riscos para projetos de software evolutivos, onde a é feita uma integração com a ferramenta CAPS (Luqi e Ketabchi, 1988). Com essa integração, métricas são capturadas automaticamente, no contexto da prototipagem evolutiva, visando a auxiliar os gerentes de projeto a melhorar a avaliação de riscos nos processos de desenvolvimento de software. Os autores propõem uma técnica de avaliação de riscos criteriosa e a ser aplicada ao final de cada ciclo de prototipagem, e argumentam que, com o uso de modelos de avaliação de riscos informais e ferramentas de controle de projetos otimistas, gerentes de projetos condenam a si mesmos a extrapolar cronogramas e custos.

4. Aspectos Práticos e Operacionais

Além da fundamentação apresentada junto aos enunciados dos requisitos, alguns aspectos devem ser considerados como um todo para a utilização das técnicas e metodologias de prototipagem evolutiva no desenvolvimento de software baseado na Web.

O desenvolvimento de sistemas deve ser visto como um processo de aprendizado para todas as partes envolvidas, não apenas a transformação de requisitos no sistema alvo (Budde e Zullighoven, 1990). A mensagem passada pelos autores não é propriamente um requisito para a utilização da prototipagem, e nem mesmo para os processos de desenvolvimento de software, mas é uma observação que, se levada em consideração pelos envolvidos no processo de prototipagem, pode auxiliá-los a alcançar seus objetivos.

Vale ressaltar que as especificações de requisitos e a prototipagem apresentam uma grande sinergia; seja na prototipagem descartável, onde os protótipos tendem a fazer parte da especificação; ou na prototipagem evolutiva, quando protótipo e especificação ficam interligados e se retroalimentam ao final de cada iteração. Schrage (1996) comenta que especificações e protótipos podem se reforçar mutuamente ou se tornarem inimigos implacáveis, e cita ainda que algumas organizações são essencialmente guiadas pelas especificações, já outras, pela prototipagem.

É importante que as organizações oficializem o uso da prototipagem para poderem aplicá-la de forma mais adequada e controlada. Conforme mencionado por Budde e Zullighoven (1990), muitas organizações aplicam a prototipagem sem lhe dar esse nome explicitamente.

5. Conclusão

O presente estudo gerou um catálogo de requisitos para a utilização de prototipagem evolutiva nos processos de desenvolvimento de software baseado na Web. Os requisitos apresentados foram retirados de trabalhos que relatam experiências com o desenvolvimento de software com o uso de metodologias, técnicas e ferramentas de prototipagem e execução ou análise de projetos, terminados com sucesso ou abandonados, por excessivas falhas ou estratégias gerenciais. Muitos dos requisitos foram repetidamente citados em diversas referências, enquanto que outros foram narrados por poucos autores. Assim, o catálogo não pretende ser definitivo nem mesmo conter todos os requisitos necessários ao uso da prototipagem evolutiva no desenvolvimento de software baseado na Web, mas deve auxiliar os usuários a comparar, selecionar e personalizar as metodologias de prototipagem disponíveis.

Referências

Budde, R., Zullighoven, H. (1990) "Prototyping revisited", Proceedings of the 1990 IEEE International Conference on Computer Systems and Software Engineering, Tel-Aviv, Israel, p. 418-427.

Buskirk, R. V., Moroney, B. W. (2003) "Extending Prototyping", IBM Systems Journal, vol. 42, No. 4, p. 613-623.

Dagnino, A. (2002) "An evolutionary lifecycle model with Agile practices for software development at ABB", Proceedings of the Eighth IEEE International Conference on Engineering of Complex Computer Systems, Raleigh, EUA, p. 215-223.

Gordon, V. S., Bieman, J. M. (1995) "Rapid prototyping: lessons learned", IEEE Software, vol. 12, No. 1, p. 85-95.

Hekmatpour, S (1987) “Experience with evolutionary prototyping in a large software project”, ACM SIGSOFT Software Engineering Notes, vol. 12, No. 1, p. 38-41.

Jeenicke, M., Bleek, W-G., and Kliscewski, R. (2003) “Revealing Web User Requirements through e-Prototyping”, Proceedings of the 15th International Conference on Software Engineering and Knowledge Engineering: SEKE 03. São Francisco, EUA.

Lichter, H., Schneider-Hufschmidt, M., Zullighoven, H. (1993) “Prototyping in industrial software projects—bridging the gap between theory and practice”, Proceedings of the 15th International Conference on Software Engineering, Baltimore, MD, USA, p. 221-229.

Luqi, Berzins, L.V., Shing, M., Riehle, R., Nogueira, J. (2000) “Evolutionary Computer Aided Prototyping System (CAPS)”, Proceedings of the 34th International Conference on Technology of Object-Oriented Languages and Systems, Santa Barbara, CA, USA, p. 363-372.

Luqi, Ketabchi, M (1988) “A Computer-Aided Prototyping System”, IEEE Software, vol. 05, no. 2, p. 66-72.

Luqi, Nogueira, J. (2000) “A risk assessment model for evolutionary software projects”, Proceedings of the 2000 Monterey Workshop on Modelling Software System Structures in fastly moving Scenario, Santa Marguerita Ligure, Itália, p. 208-215.

Ozaki, H., Ban, S., Gondow, K., Katayama, T. (2003) “An environment for evolutionary prototyping Java programs based on abstract interpretation”, Proceedings of the Tenth Asia-Pacific Software Engineering Conference (APSEC’03), Ishikawa, Japão, p. 362-370.

Paula Filho, W. P. (2001), Engenharia de Software, LTC, 2ª edição.

Scharge, M. (1996), “Cultures of Prototyping”, In *Bringing Design to Software*. T. Winograd, Ed. ACM, Nova York, NY, EUA, p. 191-213.

Walker, M., Takayama, L., and Landay, J. (2002) “High-fidelity or low-fidelity, paper or computer medium?”, Proceedings of the Human Factors and Ergonomics Society 46th Annual Meeting: HFES2002. September 30-October 4, 2002, p. 661-665.

Zhang, K., Song, G., Kong, J. (2004) “Rapid software prototyping using visual language techniques”, Proceedings of the 15th IEEE International Workshop on Rapid System Prototyping, Richardson, TX, USA, p. 119-126.