

# Adequação de Processos para Fábricas de Software

Thayssa Águila da Rocha <sup>1</sup>, Sandro Ronaldo Bezerra Oliveira <sup>1,2</sup>,  
Alexandre Marcos Lins de Vasconcelos <sup>1</sup>

<sup>1</sup>Centro de Informática – Universidade Federal de Pernambuco (UFPE)  
Caixa Postal 7851, 50732-970, Recife – PE – Brasil  
Fone/Fax: (+55 81) 2126-8430

<sup>2</sup>Centro de Ciências Exatas e Tecnologia – Universidade da Amazônia (UNAMA)  
Av. Alcindo Cacela, 287, 66060-902, Belém – PA – Brasil  
Fone: (+55 91) 210-3000, Fax: (+55 91) 225-3909  
{tar, srbo, amlv}@cin.ufpe.br

**Abstract.** *Each day the research concerning Software Factories has increased, especially due to recent term growth in the world-wide scene. Indian Factories had become quality and success reference, making that countries like Brazil came to pursue a similar model and to search results so positive how much. However, to reach this standard, it must be clear that factors as processes, quality standards and manufacture solutions frameworks intervene directly with the final result. From this point of view, this paper presents a mapping of the software development process into the different boarding of Software Factories.*

**Resumo.** *A cada dia as pesquisas acerca de Fábricas de Software vêm se intensificando, especialmente devido ao crescimento desta atividade no cenário mundial. Fábricas da Índia se tornaram referência de qualidade e sucesso, fazendo com que países como o Brasil viessem a perseguir um modelo semelhante e buscar resultados tão positivos quanto. Porém, para atingir este padrão, devemos estar cientes que fatores como processos, padrões de qualidade e frameworks de soluções fabris interferem diretamente no resultado final. Neste contexto, este artigo apresenta um mapeamento dos tipos de processo de desenvolvimento às diferentes abordagens de Fábricas de Software.*

## 1. Introdução

A exemplo do crescimento e amadurecimento das fábricas de software da Índia [Kripalani 2003], as iniciativas brasileiras têm se multiplicado e apresentado um crescimento considerável nos últimos meses [Cesar 2004], especialmente devido a fatores competitivos, uma vez que o próprio mercado nacional tem se tornado mais exigente em termos de qualidade do produto e de redução de custos [Tartarelli 2004].

Desta forma, as iniciativas de organização do modelo fabril, moldado a partir de preceitos como o *taylorismo* e o *fordismo*, vindos desde o século XIX, têm tentado mapear conceitos de produção em larga escala com qualidade para o mercado de software, aumentando a produtividade e reduzindo os custos de produção, de forma

semelhante à proposta de Taylor e Ford, no surgimento das fábricas tradicionais [Tartarelli 2004].

No entanto, o caso específico de uma fábrica de software requer uma organização mais holística, que leve em consideração vários fatores como gestão de pessoas, gestão empresarial, qualidade de software, de processos e de produtos, utilização de ferramentas, etc. Dentro deste contexto, este artigo irá focar no fator processo de desenvolvimento, fornecendo um mapeamento dos tipos de processo - quanto à sua execução - às diferentes abordagens de Fábricas de Software.

Como referência para as abordagens, ou classificações, de Fábrica de Software será utilizada a perspectiva de *escopo de fornecimento*, utilizada por Fernandes (2004).

A importância da escolha dos processos que melhor se adequem a uma iniciativa de Fábrica de Software é baseada no aumento de destaque que a definição e padronização de processos vem sofrendo, especialmente após a iniciativa do Software Engineering Institute da Universidade de Carnegie Mellon, quando da criação do CMM [Paulk 1993] – Capability Maturity Model for Software, que define níveis de capacidade para uma organização que tem a produção de software como objetivo primeiro.

Segundo Zahran apud Fernandes (2004), o processo funciona como o elo de ligação entre os outros elementos desta visão holística que norteia as Fábricas de Software, e deve interligar a Organização, o Gerenciamento, as Habilidades e a Tecnologia utilizada, pois embasa a criação dos papéis organizacionais e definição das responsabilidades, atividades e documentos (artefatos de insumo e produto), assim como as diretrizes para as práticas gerenciais e para a seleção da tecnologia que será utilizada durante a produção, ou seja, dependendo do objetivo da fábrica, a escolha do processo mais adequado é de suma importância para o sucesso da organização.

Para obter os resultados esperados, o artigo será estruturado da seguinte forma: na seção 2, o conceito de Fábrica de Software será definido e classificado. Na seção 3, os processos de desenvolvimento serão também conceituados, classificados quanto à execução e será detalhado como estes podem ser implementados e definidos em uma organização, fornecendo embasamento para a seção 4, onde a adequação destes processos será mapeada para os diferentes tipos de fábricas identificados anteriormente, assim como um breve guia de implementação é apresentado. A seção 5 finalmente apresenta a conclusão do artigo, identificando os trabalhos futuros acerca do tema.

## 2. Fábricas de Software

Segundo Bemer apud Cusumano (1989), o termo **Fábrica de Software** vem sendo discutido desde o final dos anos 60, e evoluindo e se refinando até os dias atuais.

Segundo Cusumano (1989) um processo fabril constitui-se na produção de produtos em massa, incluindo operações centralizadas de larga escala, tarefas simples e padronizadas, controles padronizados, trabalhadores especializados, mas com poucas habilidades, divisão de trabalho, mecanização e automação do processo. Desta forma, a associação do termo fábrica ao desenvolvimento de software sugere que se apliquem técnicas para produção em larga escala, de forma coordenada e com qualidade.

Desde 1968 alguns estudos publicados associam ainda características como reusabilidade, utilização de ferramentas para suportar o desenvolvimento, sistemas de controle e gerenciamento, modularização e produção de famílias de produtos como básicas para uma organização que se intitula uma Fábrica de Software.

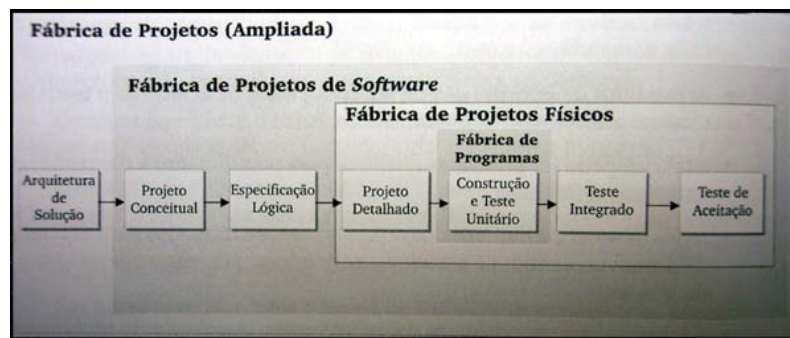
Mais recentemente, Greenfield (2003) nos apresenta uma visão semelhante, onde o conceito de Fábrica de Software está fundamentado no desenvolvimento baseado em componentes, direcionado a modelos e a linhas de produto de software que caracterizariam uma iniciativa de fábrica, visando tornar a montagem de aplicações mais barata através de reuso sistemático, possibilitando a formação de cadeias de produção.

Já Fernandes (2004) apresenta fábricas de software como *“Um processo estruturado, controlado e melhorado de forma contínua, considerando abordagens de engenharia industrial, orientado para o atendimento a múltiplas demandas de natureza e escopo distintas, visando à geração de produtos de software, conforme os requerimentos documentados dos usuários e/ou clientes, da forma mais produtiva e econômica possível”*.

Este conceito baseia-se em alguns atributos básicos que o autor advoga como imprescindíveis em qualquer Fábrica de Software, seja qual for a sua categorização. Alguns destes atributos são: processo definido e padrão (desenvolvimento, controle e planejamento); interação controlada com o cliente (entradas e saídas da fábrica); solicitações de serviço à fábrica devem ser padronizadas; estimativas de custos e prazos baseadas no conhecimento real da capacidade produtiva com métodos de obtenção baseados em dados históricos; controle rigoroso dos recursos envolvidos em cada demanda da fábrica; controle e armazenamento em bibliotecas de itens de software (documentos, código, métodos, etc); controle dos status e execução de todas as demandas; produtos gerados de acordo com os padrões estabelecidos pela organização; equipe treinada e capacitada nos processos organizacionais e produtivos; controle da qualidade do produto; processos de atendimento ao cliente; métricas definidas e controle dos acordos de nível de serviço definidos com o cliente.

Uma vez definido o conceito de Fábricas de Software e citadas algumas de suas características, as mesmas serão classificadas a fim de que possa ser caracterizado o mapeamento dos processos.

A classificação adotada será a de Fernandes (2004), onde, segundo a Figura 1, são apresentados os quatro tipos de fábricas classificadas de acordo com o seu escopo de atuação ao longo das fases de desenvolvimento de um projeto de Software.



**Figura 1. Classificação de Fábricas de software de acordo com seu escopo de fornecimento [Fernandes 2004]**

O autor também cogita a existência de uma fábrica de testes, que englobaria apenas a fase de teste integrado. Porém, para o objetivo deste artigo, a fábrica de testes se enquadraria na mesma classificação da fábrica de programas que será definida a seguir, apesar de gerarem um produto final distinto.

A **fábrica de programas** consiste na menor unidade de fábrica, conseqüentemente a menos complexa. Tem por objetivo principal codificar e testar programas de computador. No seu processo produtivo engloba praticamente as fases de construção e testes unitários.

A **fábrica de projetos** atua com um pouco mais de abrangência no processo de produção, englobando além das atividades inerentes à fábrica de programas, fases como projeto conceitual, especificação lógica, projeto detalhado da solução, realização de testes de integração e de aceitação. Dependendo da interface com o cliente, a fábrica pode se caracterizar por **projetos de software** ou **projetos físicos**, porém, seus requisitos e características básicas são muito semelhantes. No caso das fábricas de projetos de software, há a necessidade do conhecimento do negócio do cliente. A **fábrica de projetos ampliada** não será tratada neste mapeamento pois engloba soluções mais abrangentes de Tecnologia da Informação fugindo ao escopo do artigo, podendo ser representada no nível da fábrica de projetos de software.

Mesmo não estando incluída na Figura 1, o **modelo de outsourcing de sistemas** é citado como uma especialização da fábrica de projetos, onde a mesma é dedicada exclusivamente a um determinado cliente, diferenciando apenas na interface da fábrica com o cliente, que deve ser adaptada aos critérios e regras estabelecidas previamente entre ambos, normalmente através de um SLA – Service Level Agreement – que descreve estes critérios, restrições e procedimentos de mudança no escopo e na avaliação do serviço [Assano, 2002]. O modelo de outsourcing apresentado pelo autor é bem mais complexo e envolve outros processos auxiliares, que tornam-se essenciais para esta modalidade.

Merece destaque ainda a **fábrica de componentes**, porém, segundo Basili (1994) os processos de desenvolvimento atuais não suportam diretamente a utilização de componentes, uma vez que definem apenas as fases de desenvolvimento do projeto, que utilizam os componentes já catalogados. Desta forma, o mapeamento não se estenderá a esta classe de Fábricas de Software. Entretanto, os processos devem ser escolhidos de forma a utilizarem e tirarem as vantagens inerentes da reutilização de artefatos durante todo o ciclo de vida.

### 3. Processos de Desenvolvimento

Informalmente, o **processo de software** pode ser compreendido como o conjunto de todas as atividades necessárias para transformar os requisitos do usuário em software [Humphrey 1989]. Um processo de software é formado por um conjunto de passos de processo parcialmente ordenados, relacionados com conjuntos de artefatos, pessoas, recursos, estruturas organizacionais e restrições e tem como objetivo produzir e manter os produtos de software finais requeridos [Lonchamp 1993].

Os passos de processos são **atividades**. Uma atividade é um passo de processo que produz mudanças de estado visíveis externamente no produto de software. As atividades estão associadas a papéis, ferramentas e artefatos; incorporam e implementam procedimentos, regras e políticas; e têm como objetivo gerar ou modificar um dado conjunto de artefatos.

Uma atividade aloca **recursos** (por exemplo, máquinas e orçamento) e é escalonada, monitorada e atribuída a desenvolvedores (agentes), que podem utilizar ferramentas para executá-las. Um **agente** está relacionado com as atividades de um processo e pode ser uma pessoa ou uma ferramenta automatizada. Diferentes agentes terão percepções diferentes acerca do que acontece durante o processo de software. Por sua vez, **artefato** é um produto criado ou modificado durante um processo. Tal produto é resultado de uma atividade e pode ser utilizado posteriormente como matéria-prima para a mesma ou para outra atividade a fim de gerar novos produtos.

A descrição abstrata do processo de software caracteriza um **modelo de processo** de software. Vários tipos de informação devem ser integrados em um modelo de processo para indicar quem, quando, onde, como e por que os passos são realizados.

Segundo [Conradi 1994], **projeto** é a instância de um processo, com objetivos e restrições específicos. Pode-se dizer que um projeto é um esforço para desenvolver um produto de software, ou seja, envolve uma estrutura organizacional, prazos, orçamentos, recursos e um processo de desenvolvimento.

#### 3.1. Tipos de Processos de Software quanto à execução

Na literatura especializada pode-se encontrar duas frentes de processo de software no tocante à execução. A primeira, definida como um processo tradicional, ou também chamada de processo pesado, que caracteriza-se por possuir como foco principal a previsibilidade dos requisitos do sistema e o comando e o controle destes a partir de um prévio planejamento antes do início do desenvolvimento, transformando estas ações em um processo rigoroso [Pressman 2000]. Rigoroso pois a especificação de requisitos torna-se a etapa fundamental, onde todas as necessidades do cliente são definidas e documentadas, sendo que para cada um destes requisitos, são gerados outros documentos, tornando o processo de análise e projeto bastante demorado e de difícil manutenção caso alguma especificação seja alterada. Pode-se, ainda, caracterizar que este tipo de processo possui uma abordagem voltada para o planejamento detalhado, fases sequenciais de processo e artefatos de uma fase para a seguinte. Como exemplo deste tipo de processo tem-se o RUP [Krutchen 2003], o OPEN [Sellers 1997] e o Catalysis [D'Souza 1999].

Já a outra frente, chamada de processo ágil ou processo leve, possui seu foco na eficiência, abordando como premissa o compromisso entre “nada de processo” e

processos rigorosos [Beck 2000]. Esta frente propõe que a análise de requisitos seja extremamente mutável, “abraçando” mudanças como principal área de atuação da metodologia. Sendo assim, os planejamentos são constantes, não havendo uma etapa exclusiva para isso, ficando o foco principal com a codificação. Para isso os meios para estes fins são: adaptabilidade; cada item de processo deve agregar valor; orientação a pessoas; comunicação; e aprendizado. Para exemplificar esta outra linha de processos temos o XP [Jefries 2001], o Crystal [Cockburn 2002] e o SCRUM [Bach 1995].

Desta forma, podemos listar como principais pontos fracos destes tipos de processo, os listados a seguir:

- **Processo Tradicional:** a burocracia, uma vez que agrega mais tarefas para as suas ações; e a não adaptabilidade, onde a realidade (prazo, escopo, processo, pessoas) difere do planejado / documentado;
- **Processo Ágil:** a escalabilidade para equipes grandes / dispersas; e a mudança de cultura de paradigma.

Já como pontos fortes, temos:

- **Processos Tradicionais:** é baseado em workflows de processo; orientado a planejamentos; garantia alta de desenvolvimento; equipes e produtos grandes; projetado para suportar requisitos correntes e desconhecidos; modelo de desenvolvimento de software completo incluindo suporte a ferramentas; atribuições centradas no objetivo das atividades; possibilita ser instanciado e customizado de acordo com o domínio da aplicação ou da organização.
- **Processo Ágil:** revisa-se o código todo o tempo; toda a equipe irá testar o software inclusive o cliente; toda a equipe torna a atividade de projeto diária; a arquitetura será definida e refinada todo o tempo; possui interações curtas caracterizadas por minutos e horas; modularidade no nível de desenvolvimento do processo; orientado a pessoas.

#### 4. Adequando Processos a Fábricas de Software

A fim de obter um ganho maior de produtividade e facilitar o atendimento dos objetivos que a fábrica se propõe, é importante saber escolher o processo de desenvolvimento que melhor se adequa.

Os objetivos da fábrica podem ser definidos de acordo com vários fatores, dependendo do tipo de fábrica. Neste trabalho, estamos classificando as Fábricas de Software em: fábricas de código, de projetos, de componentes e outsourcing de sistemas. Já os processos de desenvolvimento foram classificados em: ágeis e tradicionais. Desta forma, o mapeamento destas classes de processos aos tipos de fábrica se torna mais simples e baseado no objetivo final da fábrica.

É importante salientar que no caso da fábrica em questão possuir meta de adoção de algum modelo de qualidade, maiores cuidados devem ser tomados acerca da escolha do processo, e não apenas as características que estamos levando em consideração, pois pode ser necessária a formalização de alguns processos de suporte que podem não ser definidos, ou ser parcialmente, pelo processo de desenvolvimento adotado, ainda que este seja um processo classificado como tradicional.

Modelos de qualidade como CMM [Paulk 1993], CMMI [Chrissis 2003] e SPICE [SPICE 2004] são mais abrangentes que os processos de desenvolvimento mais conhecidos e usados pelo mercado, pois se estendem a áreas como controle de qualidade e atendimento aos fornecedores, que não fazem parte do processo de produção da solução diretamente. Desta forma, mesmo que o processo de desenvolvimento fosse escolhido corretamente, este não seria suficiente para o sucesso total da fábrica. A inclusão das características dos modelos de qualidade está prevista como um trabalho futuro, conforme descrito na seção 5 deste artigo.

A tabela a seguir (ver Tabela 1) ilustra o mapeamento proposto, que será descrito em seguida.

**Tabela 1 - Mapeamento de Processos para Fábricas de Software**

Processos	Ágeis				Tradicionais			
Fábricas	Iterações Curtas	Pequenas Equipes	Orientado a Pessoas	Dinâmico	Garantia alta de Qualidade	Grandes Equipes	Tarefas centradas em atividades	Orientado a Planejamento
<i>Código</i>	X	X	X	X				
<i>Componente</i>	X	X	X	X				
<i>Projeto</i>						X	X	X
<i>Outsourcing</i>				X	X	X	X	X

Para a **fábrica de código**, que requer maior agilidade e pouco formalismo na documentação do produto final, os métodos ágeis poderiam ser bem utilizados, desde que respeitando os requisitos mínimos de documentação e gerência exigidos para uma fábrica de software institucionalizada. A velocidade e o dinamismo deste tipo de processo pode dar o ritmo necessário para o sucesso da operação, entretanto, deve-se manter em pequenas equipes cujos membros estejam reunidos em um mesmo ambiente físico.

A **fábrica de componente** também pode se adaptar ao pouco formalismo dos métodos ágeis, optando por documentar os componentes apenas através ferramentas como *JavaDoc* [JavaDoc 2004]. Da mesma forma que na fábrica de código, o dinamismo dos métodos ágeis também podem ser o diferencial da fábrica, porém, algum formalismo ou processo auxiliar deve ser definido para a catalogação e distribuição dos componentes gerados. As restrições de pequenas equipes cujos membros estejam reunidos em um mesmo ambiente físico também devem ser observadas.

Na **fábrica de projeto**, que se assemelha à maioria das fábricas que desenvolvem aplicações personalizadas, a necessidade dos métodos tradicionais ou aparece em maior destaque, uma vez que se faz necessário documentar e não raramente, validar formalmente com o cliente as decisões de requisitos e/ou projeto. Estes produtos também possuem um ciclo de vida maior, o que torna comum durante o

desenvolvimento de um determinado projeto a entrada e saída de pessoal, o que dificultaria a utilização de um processo orientado a pessoas como os processos ágeis.

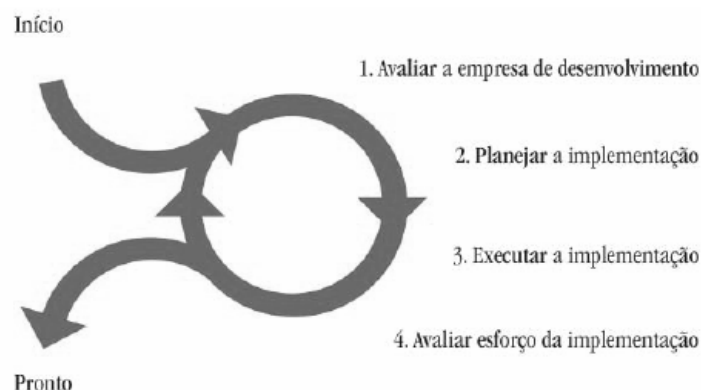
Por fim, o **outsourcing de sistemas**, que requer além de maior formalismo, maior controle por parte do cliente dos índices gerados pela fábrica – a fim de acompanhar o SLA – prática comum nesta situação, é claramente orientado a processos tradicionais. É bastante comum, e muitas vezes requerido pelo próprio cliente, a certificação da fábrica em um modelo de qualidade reconhecido mundialmente, o que reforça a necessidade da utilização de processos bem definidos e acima de tudo da coleta e acompanhamento de métricas de produção. Apesar do posicionamento adotado neste artigo, é importante citar a referência de uma experiência relatada por Fowler (2004) em uma iniciativa de implantação um processo ágil em um outsourcing de sistemas utilizando processos ágeis.

#### 4.1. Implementação de um Processo de Software

Mudar ou implantar um processo de desenvolvimento em uma fábrica de software pode ser uma tarefa difícil de se realizar e, muitas vezes leva-se tempo para ver seus resultados. Segundo Balduino [Balduino 2002], é diferente de se adotar uma nova ferramenta de desenvolvimento, já que basta instalá-la, ler o manual do usuário, seguir as instruções de um tutorial e talvez fazer um curso sobre ela. Pode-se levar algumas horas ou dias para fazer isso. Porém, mudar o processo de desenvolvimento de software de uma empresa afeta a maneira como os indivíduos trabalham, como eles vêem, e dão valor ao resultado de seu esforço.

Portanto, essa mudança não acontece da noite para o dia, por isso ela deve ser cuidadosamente planejada e gerenciada. Uma abordagem de adoção gradual do processo de desenvolvimento e ferramentas de apoio, onde cada passo seja planejado, executado e avaliado com critério, dá a sensação de se está se fazendo a implementação da maneira mais adequada.

Falando sob o aspecto da engenharia de processo, implementar um novo processo em uma empresa de desenvolvimento de software é um projeto por si só e que, por sua vez, pode ser descrito em quatro passos distintos, conforme a Figura 2.



**Figura 2. Passos para implementar processo em uma empresa [Balduino 2002]**

Os passos definidos na Figura 2 são descritos a seguir:



- **Avaliar a empresa de desenvolvimento:** o intuito é coletar informações das pessoas-chave internas ou externas à empresa para obter uma lista abrangente dos problemas atualmente encontrados, entender como essas pessoas vêem os problemas e os priorizam;
- **Planejar a implementação:** nesta fase passamos a planejar a forma como ela vai se mover do seu estado atual para onde se quer chegar. Para isto faz-se necessário uma análise dos objetivos a serem alcançados; identificar, analisar e priorizar os riscos inerentes à implantação do processo; usar um projeto piloto para avaliação inicial; estabelecer um plano de treinamento; alocar mentores como disseminadores do conhecimento;
- **Executar a implementação:** esta fase significa executar os projetos de software escolhidos para dotar o processo e as ferramentas. Do ponto de vista organizacional, isso significa: monitorar os projetos de desenvolvimento de software; gerenciar a adoção de processo nos projetos; monitorar a criação e uso de um ambiente de desenvolvimento organizacional;
- **Avaliar o esforço da implementação:** após o processo ter sido implementado no projeto piloto ou nos demais projetos, precisa-se validar os resultados contra o plano proposto no passo “Planejar a implementação”.

Em uma organização, diferentes processos podem coexistir, adequados a diferentes projetos. Para organizar e disciplinar o desenvolvimento de software é importante determinar as atividades fundamentais que deverão estar presentes em qualquer processo definido. A definição de um processo padrão estabelece uma estrutura comum a ser utilizada pela organização nos seus projetos de software e constitui a base para a definição de todos os processos [Rocha 2001]. Dessa forma, estabelece-se um processo básico que servirá como ponto de partida para a posterior definição dos processos de software adequados às diferentes características de cada projeto, permitindo economia de tempo e esforço na definição de novos processos.

A definição do processo padrão pode ser realizada tendo como base alguma norma de qualidade de processo de software e as características do desenvolvimento de software na organização. A definição poderá considerar um dos modelos de maturidade atualmente utilizados.

Tendo em vista que tipos de software diferentes possuem características distintas e requerem diferentes abordagens de desenvolvimento, o processo de software padrão da organização deverá ser adaptado (especializado) considerando-se as características relacionadas ao tipo de software (por exemplo, sistemas de informação) e ao paradigma de desenvolvimento utilizado (por exemplo, orientação a objetos).

A instanciação para projetos específicos consiste na adaptação de um processo especializado a um projeto, considerando-se as suas peculiaridades. Nesta etapa, são definidos o modelo de ciclo de vida, os métodos e as ferramentas que serão utilizadas no projeto, os recursos humanos e suas responsabilidades ao longo do processo e os artefatos (produtos) consumidos e gerados.

## 5. Considerações Finais

O conceito de Fábrica de Software está baseado na idéia de prover uma linha de produção de soluções que atendam às necessidades específicas de cada cliente. Isto se dá através da formalização de todas as atividades e seus produtos, trabalhando em forma de linha de produção, com etapas e tarefas bem definidas para cada tipo de profissional, indo das tarefas básicas da linha de produção até rotinas de controle de qualidade [Brito 2004].

Assim, com a alta especialização dos profissionais, cada um garante a produtividade da etapa de produção em que está engajado e a qualidade do artefato produzido para a etapa seguinte.

Para o perfeito funcionamento das atividades de uma Fábrica de Software é fundamental a adoção de um processo de desenvolvimento que defina as tarefas, produtos e responsáveis pelas etapas do ciclo de vida do software.

Assim, este trabalho apresentou sugestões de um possível mapeamento de tipos de processo de desenvolvimento de software, quanto à sua execução, de acordo com os tipos de Fábrica de Software, classificadas por [Fernandes 2004].

Como visões de trabalhos futuros pretende-se aprofundar este estudo em uma pesquisa que dará origem a uma monografia, acrescentando-se mais uma dimensão ao mapeamento: os modelos de qualidade e suas características, focando nos modelos mais utilizados no mercado mundial e comparando à realidade brasileira, experimentando tais modelos e suas variações também no ambiente acadêmico.

Este trabalho servirá como base para uma dissertação do programa de Mestrado do CIN/UFPE (Centro de Informática da Universidade Federal de Pernambuco), que tem como objetivo propor um modelo de fábrica de software que seja escalável, de acordo com a classificação da fábrica que o estará instanciando, e que garanta, em termos de organograma e atividades básicas, a execução de todas as exigências mínimas para adequação ao modelo de qualidade escolhido, orientando também na definição do tipo de processo de desenvolvimento a ser adotado.

## Referências

- Bach, J. (1995) “SCRUM Software Development Process - Building The Best Possible Software”, ADVANCED DEVELOPMENT METHODS.
- Balduino, R. (2002) “Implementação de um processo de desenvolvimento de software: uma abordagem passo-a-passo”, Rational Software White Paper.
- Basili, V.R., (1994) “Facts and Myths affecting Software Reuse”, IEEE, Maryland.
- Beck, K., (2000) “Extreme Programming Explained”, Addison-Wesley.
- Brito, J. A. J. (2004) “Metodologia para Gestão do Processo de Qualidade de Software para Incremento da Competitividade da Mobile”, <http://www.mct.gov.br/Temas/info/Dsi/PBQP/Reuniao%20Petropolis/Apresentacao%20Mobile.pdf>.
- Cesar, R. (2004). “Fábrica de Software: Uma vocação nacional?”, <http://www.siscorp.com.br/imprensa/computerworld02.htm?documento=24655&Area=51>, Agosto.

- Cockburn, A. (2002) “Crystal Clear – A human-powered methodology for small teams including the seven properties of effective software projects”, Humans and Technology.
- Conradi, R. et. Al (1994) “EPOS: Object-Oriented Cooperative Process Modelling”, In: FINKELSTEIN, Software Process Modelling and Technology, Kauton Research Studies Press.
- Chrissis, Mary Beth; Konrad, Mike; Shrum, Sandy. (2003). “CMMI – Guidelines for Process Integration and Product Improvement”. Boston.
- Cusumano, M. A. (1989) “The software factory: a historical interpretation”. IEEE Software, 6(2), p.23-30, Cap. 14.
- D’souza, D., Wills, A. (1999) “Objects, Components and Frameworks with UML – The Catalysis Approach”, Addison-Wesley Publishing Company.
- Fernandes, A.A., Teixeira, D. S. (2004). “Fábrica de Software: Implantação e gestão de Operações”, Atlas, São Paulo.
- Fowler, M. (2004) “Using a Agile Software Process with Offshore Development”, <http://www.martinfowler.com/articles/agileOffshore.html>, Abril.
- Humphrey, W. S. (1989) “Managing the Software Process”, New York: Addison-Wesley.
- “JavaDoc Tool” (2004) , <http://java.sun.com/j2se/javadoc/>, Agosto.
- Jeffries, R. (2001) “What is Extreme Programming?”, <http://www.xprogramming.com/xpmag/whatis.htm>, Junho.
- Kripalani, M., Engardio P., Hamm, S. (2003) . “The Rise of India”, [http://www.businessweek.com/magazine/content/03\\_49/b3861001\\_mz001.htm](http://www.businessweek.com/magazine/content/03_49/b3861001_mz001.htm), BusinessWeek Online, Dezembro.
- Kruchten, P. (2003) “Introdução ao RUP – Rational Unified Process”, Rio de Janeiro: Editora Ciência Moderna Ltda.
- Lonchamp, J. (1993) “A Structured Conceptual and Terminological Framework for the Software Process Engineering”, In: INTERNATIONAL CONFERENCE ON THE SOFTWARE PROCESS, Berlin, Proceedings.
- Paullk, M. C., Curtis, B., Chrissis, M. B., Weber, C. V. (1993) “Capability Maturity Model for Software”, Version 1.1, Technical Report CMU/SEI-93-TR-024. Software Engineering Institute - Carnegie Mellon University.
- Pressman, R. S. (2000) “Software Engineering: A Practioner’s Approach”, 5th edition. MacGraw-Hill International Edition.
- Rocha, A. R. C., Maldonado, J. C., Weber, K. C. (2001) “Qualidade de Software: Teoria e Prática”, São Paulo: Prentice Hall.
- Sellers, B. H. et. Al (1997) “The OPEN Process (Tasks, Techniques and Management)”, Chapter of Handbook of Object Technology, Centre for Object Technology Applications and Research – School of Information Technology – Swinburne University of Technology, CRC Press.

“SPICE- Software Process Improvement and Capability dEtermination” (2004),  
<http://www.sqi.gu.edu.au/spice/>, Agosto.

Tartarelli, R.V., Winckler, W. S. (2004) “Aprendizagem organizacional em fábricas de software”, <http://www.pmirs.org/Estudos/Rubens.pdf>, Agosto.