

# **Construindo uma Fábrica de Software: da Concepção às Lições Aprendidas**

**Vivianne da Nóbrega Medeiros  
Carlos Andreazza Rego Andrade  
Eduardo Santana de Almeida  
Jones Albuquerque  
Silvio Meira**

Universidade Federal de Pernambuco, Centro de Informática,  
Recife, Brasil, 50732-970  
{vnm, cara, esa2, joa, slrm}@cin.ufpe.br

e

**Eduardo Santana de Almeida  
Silvio Meira**

C.E.S.A.R. - Centro de Estudos e Sistemas Avançados do Recife  
Recife, Brasil, 50030-390  
silvio@cesar.org.br

## **Abstract**

In recent years, there have been a number of experiences with software factories reported in the literature which, although dense in a number of ways, have been lacking in what regards the discussion of the stages of definition and setting up of factories themselves. This paper discusses a number of issues related to the conception, implementation and improvement of a software factory and, as a result of a real life experiment, also points to a number of lessons learned, which can very likely be replicated within similar contexts.

**Keywords:** Software Engineering, Software Factory

## **Resumo**

Pesquisas e esforços envolvendo fábricas de software têm sido apresentados ao longo dos anos na literatura. Entretanto, ainda existe uma carência de relatos de experiências envolvendo as etapas necessárias para sua definição e construção. Assim, este artigo apresenta experiências com uma fábrica de software descrevendo desde a fase de definição, até as lições aprendidas durante todo o processo, oferecendo uma contribuição significativa neste seguimento.

**Palavras-chave:** Engenharia de Software, Fábrica de Software

## 1. Introdução

O desempenho cada vez mais otimizado das fábricas industriais clássicas, a consolidação das técnicas de engenharia de software, juntamente com o refinamento dos ambientes de desenvolvimento e o surgimento de novos ambientes de projeto e suporte integrados têm feito com que, cada vez mais, esforços sejam despendidos no sentido de realizar o conceito de Fábrica de Software [7]. Esse conceito simboliza uma desejada mudança de paradigma da produção de software focada no trabalho intensivo, para um estilo mais focado no capital, onde investimentos substanciais podem ser feitos sob um nível de risco aceitável.

As fábricas clássicas, onde as pessoas atuam como máquinas na realização de tarefas pré-determinadas, não é um modelo nem desejável, nem correto para fábrica de software. No contexto de software, a analogia com a fábrica pode ser aplicada apenas aos objetivos da produção baseada no estilo industrial, e não na sua implementação. A manufatura de software envolve pouca ou nenhuma produção tradicional: “todo sistema é único; apenas partes individuais podem aparecer repetidamente em mais de um sistema” [7].

No tocante ao conhecimento disseminado sobre fábricas de software, diversos autores [5, 6, 8, 13, 15] consideram que, muito se tem discutido sobre aspectos tecnológicos envolvendo o desenvolvimento dos produtos, enquanto que a ênfase dada sobre os processos envolvidos numa fábrica de software, principalmente, processos de definição e planejamento são negligenciados.

Neste trabalho, apresenta-se uma análise sobre as questões que devem ser consideradas durante o processo de elaboração, gestão e implantação de uma fábrica de software. Adota-se uma forma de organização para a fábrica em questão e aplica-se a mesma em duas aplicações, desenvolvidas em ambiente acadêmico, para validar a estrutura proposta, resultando no detalhamento das lições aprendidas com essas experiências, que podem ser replicadas dentro de contextos similares.

O artigo está organizado em quatro seções. A primeira seção contém esta introdução. A Seção 2 descreve o processo de criação da fábrica propriamente dito e sua implantação, sendo dividida em três subseções que detalham as lições extraídas, respectivamente, da fase de concepção, do projeto piloto e do projeto real utilizado; a Seção 3 exhibe os trabalhos relacionados e, finalmente, a Seção 4 apresenta as conclusões e perspectivas de trabalhos futuros.

## 2. Construção de uma Fábrica de Software

Nesta seção, detalham-se as etapas percorridas durante a elaboração de uma fábrica de software e sua conseqüente implantação, exibindo assim, resultados coletados com as experiências práticas vivenciadas. O modelo de fábrica proposto foi aplicado a um projeto piloto com a finalidade de gerar refinamentos e melhorias no processo adotado; em seguida, um projeto mais amplo foi desenvolvido para validar e verificar a viabilidade da estrutura definida.

### 2.1 Concepção

A fábrica foi definida em função do contexto no qual estava inserida. A idéia consistia em adotar um processo de desenvolvimento de software baseado em componentes com métodos para avaliar o progresso, técnicas para acompanhar andamento das atividades e ferramentas para otimizar a construção dos artefatos, tendo produtos desenvolvidos em um ambiente acadêmico. Devido à necessidade de incrementar a produtividade e a qualidade dos seus processos e produtos, assim como reduzir os custos e o tempo de desenvolvimento, a proposta de solução considerava dois importantes pontos: melhorar a eficiência dos processos e reutilizar os artefatos previamente construídos.

Dessa forma, durante esta etapa de concepção, que se estendeu por quatro (4) semanas, atividades relacionadas à forma de organizar a estrutura geral de uma fábrica de software foram consideradas, tendo em vista os objetivos já mencionados. As principais decisões tomadas para possibilitar a construção desse ambiente foram:

- **Perfis funcionais** e as respectivas **atividades** a serem desempenhadas;
- **Metodologia de desenvolvimento de software** a ser utilizada incluindo artefatos e **métricas**;
- Definir um **plano de processos** descrevendo as atividades e relacionando-as com os artefatos e perfis funcionais responsáveis pela execução das mesmas;
- **Material de instrumentação** necessário.

A seguir, as soluções adotadas pela fábrica proposta para contemplar os aspectos acima citados são apresentadas.

#### 2.1.1 Perfis Funcionais

A classificação estabelecida para os perfis funcionais, juntamente com as responsabilidades de cada cargo:

- **Gerente de Negócios:** prospecção do mercado e venda dos serviços.

- **Gerente de Projeto:** gerenciamento dos riscos e das atividades em desenvolvimento devendo dimensionar e alocar os recursos necessários para realização das tarefas de forma satisfatória, além de interagir com o cliente e o gerente de negócios.
- **Analista de Sistemas:** levantamento de requisitos, análise, definição da arquitetura e documentação do sistema a ser desenvolvido.
- **Analista de Qualidade:** revisão dos artefatos gerados, controle de mudanças, bem como a definição e validação da qualidade e acurácia do processo utilizado pela fábrica.
- **Engenheiro de Software:** implementação do sistema conforme as especificações e documentação, seguindo o processo de desenvolvimento definido.
- **Engenheiro de Testes:** desenvolvimento, validação e execução de testes de software com o intuito de assegurar a qualidade e acurácia do software produzido.
- **Líder de Equipe:** coordenação e atribuição de tarefas dentro de um grupo específico, relatando periodicamente ao gerente de projetos o andamento das atividades.

### 2.1.2 Metodologia de Desenvolvimento

A metodologia de desenvolvimento baseou-se no Rational Unified Process (RUP) [9] e no Project Management Body of Knowledge (PMBOK) [4]. A essência do RUP consiste em um ciclo de desenvolvimento iterativo e incremental, assim, ao longo do tempo são executadas várias iterações e cada uma delas gera versões contemplando um número maior de funcionalidades. Por sua vez, o PMBOK é aceito como um padrão para gerenciamento de projetos e descreve práticas tradicionais e largamente aplicadas para entendimento e execução das atividades relativas a gestão do projeto, incluindo objetivos, responsáveis e participantes de cada processo.

A metodologia de desenvolvimento adotada na fábrica está dividida nas seguintes fases:

- **Comercial:** definição do projeto, de acordo com as necessidades levantadas junto ao cliente, além do levantamento de estimativas de custo e esforço baseando-se na técnica de Pontos de Caso de Uso Ajustados [10]. A intenção é unificar o entendimento do produto a ser desenvolvido e fornecer uma estrutura com previsões razoáveis de recursos, custos e prazos; dados históricos devem ser coletados para utilização em estimativas de projetos futuros, aumentando assim as chances de sucesso.
- **Planejamento e Gerenciamento:** elaboração do plano do projeto (plano de trabalho, riscos, acompanhamento e controle), estimação de prazos e execução das atividades recorrentes para avaliar o progresso, coletando as métricas de desempenho ( $\mu$  concluído,  $\mu$  atraso médio e  $\mu$  novas atividades) definidas em [12]. O gerente de projetos terá atribuições e responsabilidades descritas no PMBOK. A meta é fornecer uma linha básica de estruturação e delimitação do trabalho, devendo produzir ações como: comunicar o escopo e os recursos a todos os envolvidos; definir riscos e sugerir técnicas para evitá-los, ou minimizá-los; elaborar cronograma com divisão de trabalho e dependência entre as atividades; gerenciar mudanças; planejar o acompanhamento das tarefas existentes e registrar o monitoramento através de relatórios, a fim de checar o planejado com o realizado efetivamente. Enfim, desenvolve-se mecanismos para avaliar o progresso, organizar o pessoal que desenvolverá o produto, além de rastrear e controlar o projeto e mudanças que por ventura apareçam.
- **Desenvolvimento de Componentes:** definição do problema, especificação, projeto e implementação dos componentes. Esta fase segue uma abordagem definida em [1], considerando a reutilização como fator primordial e, assim, objetivando mais qualidade e redução nos custos de desenvolvimento, testes, documentação e manutenção. Inicialmente, a abordagem parte dos requisitos do domínio do problema e produz os componentes implementados numa linguagem orientada a objetos. Uma vez implementados, o engenheiro de software desenvolve as aplicações reutilizando os componentes previamente construídos.
- **Testes e Validação:** elaboração de testes para validar os artefatos previamente construídos. Representa a última oportunidade de detectar erros antes do software ser distribuído aos usuários e tem como principais objetivos: planejar os testes que devem ser executados em cada iteração; verificar a correta integração entre todos os componentes do software; averiguar se todos os requisitos do sistema foram corretamente implementados; executar vários testes para comparar o resultado dos mesmos com os parâmetros definidos como esperados, a fim de produzir uma indicação da qualidade e da confiabilidade do software; além de realizar os testes de aceitação junto ao cliente.

### 2.1.3 Plano de Processos

O plano de processos detalha cada fase da metodologia, mostrando o fluxo de atividades a serem executadas, os artefatos que devem ser produzidos e os perfis funcionais encarregados da geração de tal material. A Tabela 1 mostra as atividades descritas no plano de processos da fábrica.

**Tabela 1. Fluxo de atividades definidas no plano de processos**

	<b>Atividades</b>	<b>Responsável</b>	<b>Artefatos</b>
<b>Comercial</b>	Levantar Necessidades do Cliente	Gerente de Negócios	- Ata de Reunião
	Elaborar Proposta Técnica	Líder de Equipe	- Proposta Técnica
	Estimar Esforço do Projeto	Gerente de Negócios	- Planilha de Estimativa de Esforço
	Elaborar Proposta Comercial	Gerente de Negócios	- Proposta Comercial
<b>Planejamento e Gerenciamento</b>	Elaborar Plano de Projetos Preliminar	Gerente de Projeto	- Plano de Projeto Preliminar
	Definir o Controle do Projeto	Gerente de Projeto Equipe de trabalho	- Work Breakdown Structure (WBS) Cronológico - Definição do Plano de Acompanhamento e Controle - Plano de Gerenciamento de Impactos - Plano de Gerenciamento de Configuração - Plano de Comunicação
	Acompanhar e Gerenciar o projeto	Gerente de Projeto Analista de qualidade	- Work Breakdown Structure (WBS) Cronológico atualizado - Plano de Acompanhamento e Controle atualizado - Plano de Gerenciamento de Impactos atualizado - Plano de Gerenciamento de Configuração atualizado - Formulário de Controle de Mudanças
	Comunicar através de Reuniões Periódicas	Gerente de Projeto Líderes de equipes	- Ata de Reuniões
	Validar o Projeto	Analista de Qualidade Gerente de Negócios Gerente de Projeto Líderes de equipes	- Formulário de Validação do Cliente - Documento Avaliando o Processo Adotado
<b>Desenvolvimento de componentes</b>	Definir Problema	Analista de sistema	- Mind-maps - Modelo de colaboração - Modelo de casos de uso - Documento de Requisitos
	Especificar Componentes	Analista de sistema	- Modelo de tipos - Framework de modelos - Aplicação do framework - Refinamento dos diagramas de iteração
	Projetar Componentes	Analista de sistema	- Modelo de classes - Refinamento dos diagramas de iteração
	Implementar Componentes	Engenheiro de software	- Código gerado
<b>Testes e Validação</b>	Elaborar Plano de Testes	Engenheiro de testes	- Plano de Testes
	Implementar Testes	Engenheiro de software	- Componentes de Teste
	Executar Testes	Engenheiro de testes	- Registro dos resultados
	Avaliar Testes	Engenheiro de testes	- Relatório de Avaliação de Testes
	Executar Testes de Aceitação	Usuário validador	- Observações do validador

Cada atividade descrita no plano de processos expõe as informações de quais são os seus objetivos, insumos necessários à sua realização (entradas), seqüência de passos a serem executados, resultados gerados (saídas) e perfis responsáveis pela condução da mesma. Modelos foram elaborados para cada documento definido, com o intuito de servir como roteiro durante a construção do artefato e, assim, diminuir o esforço de criação dos mesmos, garantir a presença de informações relevantes, além de facilitar a verificação de aderência à metodologia.

### 2.1.4 Material de Instrumentação

As tecnologias empregadas pela fábrica foram definidas de acordo com as categorias:

- Ferramentas de desenvolvimento e modelagem;
- Ferramentas para relatar bugs durante o desenvolvimento;
- Ferramentas de gerenciamento de projetos;
- Ferramentas para comunicação entre os participantes e disseminação do conhecimento; e
- Sistema gerenciador de banco de dados.

## 2.2 Calibração de Fábrica: Projeto Piloto

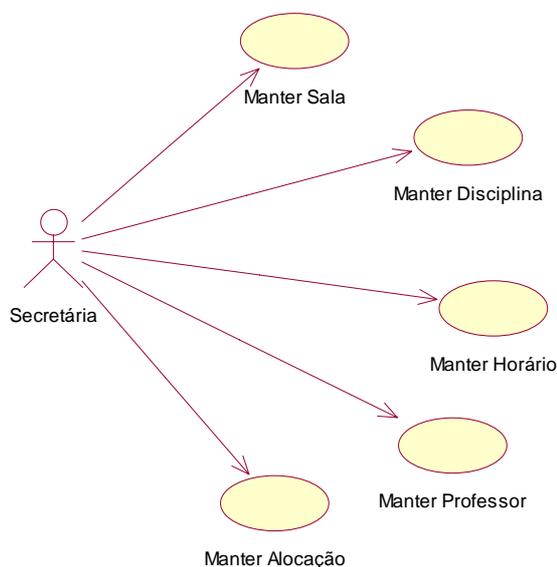
### 2.2.1 Contexto

Com o objetivo de validar e ajustar a fábrica definida, um projeto piloto foi realizado em cinco (5) semanas. Este projeto não foi complexo, já que o mais importante seria o “aculturamento” dos participantes nos processos, atividades, artefatos e tecnologias envolvidas, como também possibilitar a identificação de possíveis inconsistências durante o processo e evitar o seu posterior impacto na fase de produção, ou seja, a finalidade principal desta etapa seria ambientar os envolvidos com a estrutura utilizada, resultando em sugestões de mudanças quando necessário.

Os participantes foram alocados nos perfis apresentados na Seção 2.1.1, considerando a conveniência e o conhecimento de cada indivíduo. Entretanto, todos desempenharam mais de um perfil, a fim de evitar a ociosidade e melhorar o desempenho geral da fábrica. Os envolvidos formavam um total de treze pessoas e constituía-se de alunos dos cursos de Mestrado e Doutorado do Programa de Pós-Graduação em Ciência da Computação, da Universidade Federal de Pernambuco, que cursavam uma disciplina de Engenharia de Software, além dos dois instrutores da disciplina que desempenhavam o papel de clientes.

### 2.2.2 Execução

O projeto constituiu-se no desenvolvimento de um sistema de alocação de disciplinas, professores, salas e horários, para auxiliar os funcionários da Secretaria de Pós-Graduação em Ciência da Computação. As funcionalidades contempladas estão ilustradas num diagrama de casos de uso na Figura 1 e consistiam basicamente da manutenção (cadastro, atualização e remoção) de salas, disciplinas, horários, professores e alocações.



**Figura 1: Diagrama de casos de uso desenvolvidos no Projeto Piloto**

As etapas estabelecidas no plano de processo foram realizadas e os relatórios gerados semanalmente mostrou-se um fator essencial para o bom acompanhamento do progresso das atividades. As métricas de desempenho coletadas foram às definidas em [12] e contemplam o escopo de tempo definido para a avaliação, apresentando-se sob as seguintes variáveis:

- $\mu$  *concluído* :o quanto uma determinada equipe concluiu das atividades que estavam inicialmente planejadas;
- $\mu$  *atraso médio* :o percentual de atraso médio por atividade; e
- $\mu$  *novas atividades* :o percentual médio de tempo gasto com atividades não planejadas que alocam o pessoal da equipe, prejudicando o planejamento inicial e o desempenho. A partir desses dados, o cliente mantinha-se informado a respeito do trabalho desenvolvido e dispunha de meios práticos para avaliar a evolução do projeto.

De modo a garantir a qualidade do software piloto desenvolvido e os futuros sistemas produzidos pela fábrica, elaborou-se uma auditoria interna com o intuito de catalogar as boas práticas, desvios no processo, oportunidades de aperfeiçoamento, entre outros. Assim, foi verificado se o projeto estava seguindo a política de desenvolvimento estabelecida, a capacidade e necessidade de treinamento dos funcionários, atividades de gerência de requisitos, além de planejamento e acompanhamento de projetos e gerência de configuração. Ações de melhoria contínua, baseadas nos resultados analisados, puderam ser endereçadas.

A quantidade de horas utilizadas por fase durante a execução do piloto foram registradas e são apresentados na Figura 2.

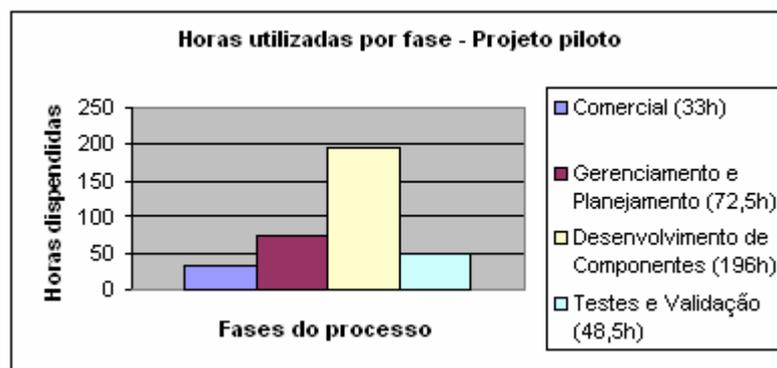


Figura 2: Horas utilizadas por fase no piloto

### 2.2.3 Análise

O projeto piloto foi realizado sem maiores problemas e o sistema foi implementado com o escopo requisitado e o prazo previsto. Esse projeto foi de extrema importância, pois permitiu aos participantes adquirir experiência na metodologia definida, verificar a adequação dos participantes em relação aos perfis funcionais e experimentar todo o processo operacional de uma fábrica de software. Entretanto, alguns pontos apresentados durante a realização do projeto piloto merecem ser destacados, a saber:

*i. Sobrecarga de Trabalho:* durante a execução do projeto, houve sobrecarga de trabalho relacionado à equipe de desenvolvimento. Este fato ocorreu, principalmente, pela baixa interseção dos horários de disponibilidade entre os membros da fábrica por se tratarem de alunos de Pós-Graduação com outras responsabilidades além da disciplina de Engenharia de Software, levando, muitas vezes, a falhas de comunicação;

*ii. Alocação de Perfis:* alguns participantes não estavam bem alocados em relação ao perfil funcional que se encontravam causando, deste modo, alguns atrasos de cronograma;

*iii. Instrumentação:* a ferramenta CASE adotada inicialmente para modelagem do sistema [2] não estava atendendo da forma esperada, uma vez que não permitia o trabalho colaborativo entre os participantes, atrasando, de certa forma, as tarefas dos membros da equipe de Analistas de Sistemas;

*iv. Avaliação do processo:* ao fim do piloto, o processo como um todo foi avaliado por todos os componentes da fábrica, através de questionários onde a eficácia do mesmo foi analisada. Durante uma reunião, após o término do piloto, foi feita uma avaliação dos pontos fortes e fracos, possibilitando assim, ajustes e melhorias em projetos subsequentes. Foi possível também monitorar o trabalho dos integrantes da fábrica através da utilização de *timesheets*, onde dados-históricos foram coletados, permitindo assim, o conhecimento das horas consumidas por fase;

*v. Estimativas:* o multiplicador de Ponto de Caso de Uso Ajustado (PCUA) para Homem/Hora (HH) utilizado durante o piloto foi 20, como sugerido pelo criador do método em [10], para empresas que não possuem dados de projetos anteriores. Dessa forma, resultou-se em uma estimativa de 445HH, porém, o total de tempo realmente gasto foi 350 horas.

Em virtude destes fatores, e pela própria execução do piloto, a fábrica sofreu alguns ajustes:

*i. Realocação de Perfis:* foi realizado a realocação de três participantes em relação aos perfis funcionais, para melhor adequação aos futuros projetos;

*ii. Ajustes de Métricas:* baseando-se nos dados históricos coletados, o multiplicador de Ponto de Caso de Uso Ajustado (PCUA) para Homem/Hora (HH) foi modificado de 20 para 18;

*iii. Instrumentação:* a ferramenta CASE utilizada foi substituída em razão das limitações encontradas.

## 2.3 Estudo de Caso: Projeto Real

### 2.3.1 Contexto

Após a estrutura proposta ter sido avaliada com a utilização de um projeto piloto, que possibilitou a coleta de métricas e resultou em ajustes para melhorar a organização geral da fábrica, um novo projeto foi desenvolvido. Visto que os participantes já tinham dados históricos e dispunham de um processo refinado, resolveu-se implantar uma maior rigorosidade, firmando um compromisso entre a prestadora de serviços (fábrica em questão) e o cliente, através da adoção de um Acordo de Nível de Serviço (SLA) [11]. Este define em maiores detalhes, e, com clareza, o relacionamento entre as partes, deixando explícito as obrigações de cada um nas interações entre as empresas e quais as penalidades que devem ser aplicadas caso o acordado não seja cumprido.

### 2.3.2 Execução

O projeto desenvolvido foi o do software SI Alocação Plus, sendo que o prazo acordado para entrega foi de seis (6) semanas. O objetivo do sistema foi o de solucionar o problema de alocação de salas (aulas, laboratórios, auditórios), professores, alunos e disciplinas, de forma automática e contemplando a eliminação de conflitos, utilizando técnicas de alocação de recursos. Este problema é conhecido na literatura como Timetabling e foi resolvido utilizando-se algoritmos genéticos [3].

O Acordo de Nível de Serviço estabelecido considerava como pontos de cobrança os seguintes indicadores:

- **I1.** Quantidade de problemas ocorridos no projeto e que não estavam registrados como riscos e/ou não tinham ações de contingência previstas;
- **I2.** Soma dos dias de atraso/adiantamento nas entregas de artefatos;
- **I3.** Quantidade de relatórios semanais de status do projeto que não foram enviados, ou que não estavam completos em relação a eventos relevantes.

Além disso, existiu uma tolerância em relação ao valor dos contadores, desde que eles não ultrapassassem a meta, que consistia na manutenção do saldo (bônus + penalidades) zerado, aplicando-se a seguinte fórmula para avaliação:  $\text{Saldo} = (I1-1) + (I2-3) + (I3)$ . Como foi a primeira vez que este tipo de acordo foi aplicado, combinou-se que os valores dos indicadores e as suas compensações seriam registrados para efeito de aprendizado, mas não aplicados.

As atividades relacionadas à garantia de acurácia dos artefatos foram efetuadas com maior rigor e constituíam-se na validação, por duas instâncias, dos documentos gerados: líder de equipe e analistas de qualidade; dessa forma, tinha-se uma avaliação mais criteriosa e, conseqüentemente, o produto final disponibilizado apresentava uma maior qualidade.

A Figura 3 apresenta os dados coletados, como a quantidade de horas utilizadas por fase durante a execução do projeto.

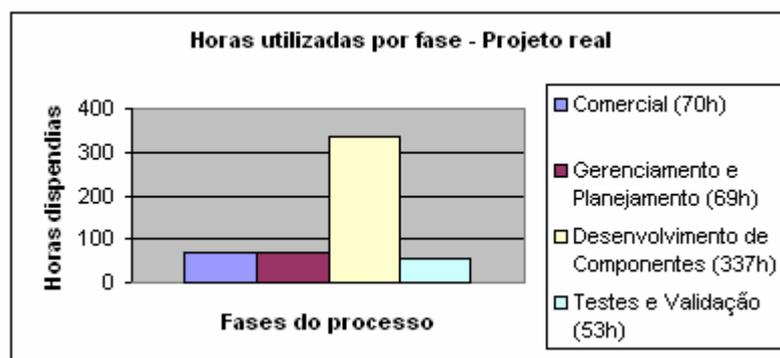


Figura 3: Horas utilizadas por fase no projeto real

As demais etapas foram similares às desenvolvidas durante o projeto piloto, com a vantagem de que a execução deste foi bastante favorecida devido à experiência adquirida durante o primeiro, resultando numa uniformização maior do número de horas trabalhadas e evitando a sobrecarga dos membros da equipe de desenvolvimento.

### 2.3.3 Análise

Ao usufruir o aprendizado adquirido com a realização do projeto piloto, algumas atividades foram realizadas de melhor modo, destacando-se as seguintes:

**i. Estimativas:** com o multiplicador ajustado, as horas estimadas foram 565 e o tempo realmente gasto foi 541 horas, ou seja, o esforço estimado mostrou-se bem mais próximo ao real do que anteriormente, isto demonstra um bom planejamento e gera segurança para estimar valores futuros;

**ii. Realocação de perfis:** o novo cenário mostrou-se vantajoso, visto que as atribuições estavam mais adequadas às habilidades dos participantes, resultando em uma maior eficiência e produtividade. A atividade de liderança nas equipes foi realizada de forma mais efetiva, permitindo um maior controle sobre a evolução de cada grupo específico;

**iii. Maturidade do processo:** após refinamento e melhor conhecimento dos membros envolvidos acerca do processo utilizado, foi possível segui-lo de maneira mais direta.

Um ajuste importante ainda se faz necessário:

**i. Evolução do SLA:** um acordo de nível de serviço deve estar bem entendido e aculturado para todas as partes antes de ser implantado, para que ele não se transforme em um problema e cumpra o seu papel. Dessa forma, recomenda-se evoluir de forma não brusca e tentar seguir a evolução do relacionamento entre os envolvidos. As alterações devem ser decididas entre as partes e apenas entram em vigor para os novos projetos que se iniciarem após essas decisões consensuais.

## 3. Trabalhos Relacionados

Alguns trabalhos envolvendo fábricas de software existem na literatura. Nesta seção, apresentaremos alguns deles, enfatizando suas diferenças e similaridades em relação ao presente trabalho.

Em [14], os autores apresentam uma vasta coleção de dados de entrevistas, questionários e métricas de software, a fim de descrever como criar e manter uma organização efetiva, centrada em processos de desenvolvimento de software.

Fernström [7] descreve o projeto *Eureka Software Factory* (ESF), que tem por objetivo permitir a criação de um mercado para produtos CASE, que possam ser configurados para aplicações específicas, ajudando os construtores de fábricas a integrar os produtos e informações utilizadas ao longo do processo.

A ESF representa um esforço conjunto de treze entidades parceiras de cinco países europeus, entre as quais encontram-se instituições de pesquisa, fabricantes de computadores, desenvolvedores de ferramentas CASE e sistemas.

A grande diferença deste trabalho em relação aos demais, está no relato das experiências com uma fábrica de software, desde a sua fase de definição até as lições aprendidas durante todo o processo.

## 4. Conclusões e Trabalhos Futuros

Neste artigo, apresentamos o processo de criação de uma fábrica de software onde foram discutidas, detalhadamente, as etapas de definição e implantação, ressaltando as lições aprendidas durante a execução dos projetos experimentais.

A execução dos projetos experimentais foi de grande importância para a apreciação do processo utilizado pela fábrica, permitindo a realização de ajustes necessários, como realocação de participantes a perfis apropriados. O êxito conseguido durante a instanciação da fábrica mostrou a viabilidade do processo adotado, além de servir como fator de motivação para realização de outros projetos.

A principal contribuição deste trabalho está numa detalhada descrição de como criar uma fábrica de software sem muitos recursos físicos e financeiros, aspecto este, não mencionado com muita frequência pela literatura.

Como trabalhos futuros, pretende-se realizar projetos de grande porte, a fim de realizar alguns ajustes e refinar ainda mais os processos anteriormente definidos, visando sempre a melhoria contínua.

## Agradecimentos

Os autores desejam agradecer o apoio de todos os integrantes envolvidos na construção e implantação da Fábrica de Software apresentada neste trabalho, a qual foi resultante de um projeto requerido pela disciplina Engenharia

de Software oferecida aos alunos da pós-graduação do Centro de Informática na Universidade Federal de Pernambuco.

## Referências

- [1] Almeida, E, S., Bianchini, C, P., Prado, A, F., Trevelin, L, C., *IPM: An Incremental Process Model for Distributed Component-Based Software Development*, In The 5th International Conference On Enterprise Information Systems (ICEIS), Angers - France. ACM Press, 2003.
- [2] Almeida, E, S.; Lucrédio, D; Bianchini, C, P.; Prado, A, F.; Trevelin, L, C. Ferramenta MVCCase - Uma Ferramenta Integradora de Tecnologias para o Desenvolvimento de Componentes Distribuídos. In: Simpósio Brasileiro de Engenharia de Software (SBES) - Sessão de Ferramentas, 2002, Gramado/RS - Brazil.
- [3] Caldeira, P., Agostinho, R., *School Timetabling Using Genetic Search*, Practice and Theory of Automated Timetabling, Toronto, 1997.
- [4] Cleland, D., *A Guide to the Project Management Body of Knowledge*, Project Management Institute, 2000.
- [5] Coulter, N., Dammann, J, *Current practices, culture changes, and software engineering education*. Computer Science Education, 5, 2, 211 – 227, 1994.
- [6] Dawson R.J., Newsham, R.W. and Kerridge, R.S, *Introducing new software engineering graduates to the 'real world' at the GPT company*. Software Engineering Journal, Vol. 07, No. 03, May, 1992.
- [7] Fernström, C., Närfelt, K., Ohlsson, L., *Software Factory Principles, Architecture, and Experiments*, IEEE Software, Vol. 09, No. 01, January, 1992.
- [8] Gibbs N., *The SEI education program: The Challenge of Teaching Future Software Engineers*, Communications of the ACM, Vol. 32, No. 05, May, 1989.
- [9] Jacobson, I., et. al., *The Unified Software Development Process*, Addison-Wesley, 2001.
- [10] Karner, G., *Resource Estimation for Objectory Projects*, Copyright Rational Software, 1993.
- [11] Karten, N. *Establishing Service Level Agreements*, 2001. In <http://www.nkarten.com/>, último acesso em maio de 2004.
- [12] Meneses, J. B., Moura, H. P., *Inspector: Um Processo de Avaliação de Progresso para Projetos de Software*, Simpósio Brasileiro de Engenharia de Software, Rio de Janeiro, 2001.
- [13] Shaw M., *Prospects for an Engineering Discipline of Software*. IEEE Software, Vol. 07, No. 06, November, 1990.
- [14] Siy, H, P., Mockus, A., Herbsleb, J, D., Krishnan, M., Tucker, G, T., *Making The Software Factory Work: Lessons from a decade of experience*, In the Seventh International Symposium on Software Metrics, London, England, April 2001.
- [15] Wasserman A.I. *Toward a Discipline of Software Engineering*. IEEE Software, Vol. 13, No. 06, November 1996.