Desenvolvimento Distribuído de Software com Scrum

José Augusto Fabri, Alexandre L´Erario, André L. dos Santos Domingues

Programa de Pós Graduação em Informática – Universidade Tecnológica Federal do Paraná (UTFPR)
Campus Cornélio Procópio – Brazil

fabri@utfpr.edu.br, alerario@utfpr.edu.br, anddomingues@utfpr.edu.br

Abstract. *This paper combines Global Development Software (GDS) and Scrum framework (GDScrum). The authors validate this propose by of the controlled experiment. The experiment show that the GDScrum can to be used by software companies whit success.*

Resumo. *O objetivo deste trabalho é aliar as prerrogativas delineadas pela teoria de desenvolvimento distribuído de software ao framework Scrum, gerando assim o DDScrum. Após propor um modelo que una teoria e framework, os autores do trabalho realizaram um experimento controlado do DDScrum. Durante o experimento foi possível verificar que empresas que desejam produzir de forma distribuída podem perfeitamente utilizar o Scrum para isto.*

1. Introdução

A alta carga tributária, a deficiência na formação da mão de obra na área de tecnologia da informação, e a ineficiência do setor produtivo caracterizam como fonte inibidora no processo de expansão externa do Brasil no setor produtivo de software. Universidade, empresa e governo devem desenvolver mecanismos que alterem este cenário.

Com base neste contexto e com o intuito de propor uma alternativa ao processo de produção de software, focando em um aspecto produtivo mais eficiente, este artigo tem como objetivo verificar se é possível aliar as prerrogativas delineadas pelo desenvolvimento distribuído de software aos processos ágeis – basicamente o Scrum.

Para atingir o objetivo proposto este trabalho foi estruturado da seguinte forma: a Seção 2 apresenta alguns conceitos sobre Scrum. A Seção 3 foca a ideia de desenvolvimento distribuído de software. O modelo que une Scrum às prerrogativas do desenvolvimento distribuído de software é mapeado na Seção 4. A Seção 5 apresenta os métodos e os procedimentos utilizados para validar, parcialmente, o modelo. A Seção 6 apresenta a aplicação do modelo por meio de um experimento controlado. Por fim, a Seção 7 apresenta a análise dos resultados, conclusões e perspectivas futuras.

2. Scrum

Segundo Sutherland (2007), o Scrum destaca-se dentre os diferentes métodos ágeis, com uma abordagem enxuta de desenvolvimento. O método surgiu em 1986, quando Takeuchi e Nonaka realizaram um estudo e notaram que pequenos projetos que tinham equipes pequenas e multifuncionais obtinham os melhores resultados. Este estudo serviu como base para que, em

1993, Jeff Sutherland, John Scumniotales e Jeff McKenna criassem o Scrum. Ken Schwaber formalizou a definição de Scrum e ajudou a implantá-lo no desenvolvimento de software em todo o mundo. Uma visão abrangente do SCRUM pode ser visualizada por meio da Figura 1.

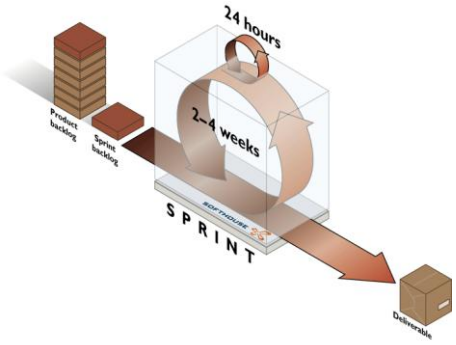

Figura 1 – Scrum *Framework*[1]

O Scrum possui um conjunto de artefatos, entre eles destacam-se: a *Story* (Estória): é uma breve descrição de uma necessidade (ou requisito) do cliente; o *Product Backlog*: são as estórias pendentes. Os requisitos que devem ser implementados para a caracterização do produto; o *Sprint Backlog*: a interpretação técnica de *backlog* do projeto, que resume as tarefas que serão feitas no decorrer do desenvolvimento pela equipe; o *Sprint*: é um período de tempo que pode variar de duas semanas até um mês que resulta em uma parte do software pronto. É importante salientar que após a execução de uma *Sprint* é realizada uma reunião entre os envolvidos com o processo.

Além dos artefatos o Scrum possui alguns papeis: o *Scrum Master*: responsável por cuidar das atualizações diárias do Scrum, papel equivalente ao de gerente de projeto; o *Scrum Team:* uma equipe composta de desenvolvedores, *DBAs* e *quality assurance testers* (responsáveis por desenvolver o produto final); *Product Owner:* voz do cliente na equipe, responsável por manter o foco do projeto nos negócios, ficando em constante contato com os clientes e futuros usuários do sistema.

3. Desenvolvimento Distribuído de Software

Desenvolvimento Distribuído de Software (DDS) é definido pela colaboração e cooperação entre departamentos de organizações e pela criação de grupos de desenvolvedores que trabalham em conjunto, porém distantes fisicamente (Prikladnicki e Audy 2006).

De acordo com Marquardt e Horvath (2001), DDS é composto por equipes globais, grupos de pessoas de diferentes países que trabalham em conjunto no desenvolvimento de um projeto. Karolak (1998) aborda o ambiente distribuído como o resultado da união de organizações virtuais, que se referem a entidades que desenvolvem partes de um projeto em locais dispersos, porém encarando-o como um projeto local.

Prikladnicki e Audy (2006) categorizam o DDS de acordo com a distância geográfica entre os sites (cada unidade de trabalho), classificando-o em níveis de dispersão nacional,

[1] Figura retirada de: http://dojofloripa.wordpress.com/2007/02/07/scrum-em-2-minutos/

continental e global, com equipes localizadas em um mesmo país, em países diferentes situados em um único continente e com alocações em mais de um país, respectivamente. Os dois últimos casos caracterizam desenvolvimento global de software (GSD).

Para considerar um ambiente como distribuído, os autores deste trabalho, apontam alguns requisitos: o ambiente distribuído deve ser constituído por no mínimo dois sites; e deve haver distância física entre eles; há uma granularidade de repasse, que identifica a forma na qual o site entrega seu subproduto para a rede; difusão do processo, aspectos processuais em comum entre os sites da rede; e grau de interação, que revela o quanto a informação é trocada entre os nós, mediante a quantidade de estímulos enviados entre uma instância e outra para realizar uma tarefa.

4. Modelo proposto - DDSCrum

Um modelo caracteriza-se como um artefato mental, destinado a mapear uma determinada realidade, ou alguns de seus aspectos, a fim de torná-los descritíveis qualitativa e quantitativamente e, em alguns casos, passíveis de observação. O surgimento dos modelos busca extrapolar a limitação da percepção humana com relação ao mundo.O modelo proposto neste trabalho reuni as características relevantes sobre a produção distribuída de software, delineadas por L´Erário (2009) e o *framework* Scrum (vide seção 3).

L´Erario (2009) caracteriza a produção distribuída em estados e elementos que indicam a dinâmica produtiva de um produto caracterizado como software (vide Figura 2). Os estados e elementos se relacionam diretamente com o produto e com a unidade de produção de software (fábricas de softwares, *software house*, institutos de pesquisa). Neste texto estas unidades são delineadas como *sites* de Produção.

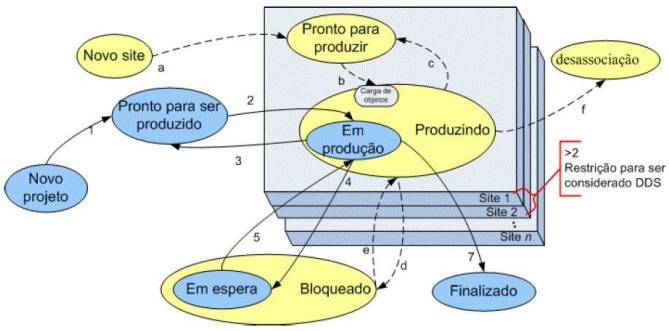

Figura 2 – Modelo de produção distribuída de software (retirado de L´Erario 2009)

Segundo L´Erario (2009), na Figura 2, o estado *Novo site* indica que um novo nó da rede produtiva está pronto para associar-se. Já o estado *Pronto para produzir*, caracteriza que o *site* já se encontra na rede, ou seja, um acordo produtivo entre dois ou mais *sites* foi celebrado.

Assim que uma ordem de serviço (conjunto de requisitos ou componentes de software) for colocada em produção, *site* pode atingir o estado *Produzindo*. Perceba que o *site* também pode transitar do estado de *Produzindo* para *Pronto para Produzir,* este fato ocorre quando existe possibilidade do *site* atender novas demandas de produção. O *site* pode atingir o estado de *Bloqueado*, caso ele não tenha possibilidade de atender mais aos critérios de qualidade (de produção) impostos no acordo celebrado pela rede; não conseguir atender demandas tecnológicas ou contratuais (possibilidade da entrega de um determinado produto (ou subproduto) dentro dos prazos e custos pré-estabelecidos) de um novo produto de software ou

estiver aguardando um artefato de produção para continuar o seu trabalho (relação de dependência de artefatos). É importante salientar que o *site* pode transitar do estado *Bloqueado* para *Produzindo*, caso ele atenda novamente aos critérios de qualidade impostos ou possuir condições de suprir as demandas contratuais ou tecnológicas de um determinado projeto. O *site* pode solicitar a sua *Desassociação* da rede, por meio, do término ou quebra do acordo.

Além de mapear estados e elementos de um *site*, a Figura 2 também caracteriza estados e elementos do projeto (de software). Dentro deste contexto é possível perceber que um *Novo projeto* quando concebido é fatiado em ordens de serviços (OS)[2] e atingi o estado *Pronto para ser produzido*. As ordens de serviços são direcionadas a um determinado *site* e colocadas *Em produção*. Assim que a ordem de serviço é concluída o estado *Finalizado* é caracterizado. É factível que a produção de uma determinada OS (a) dependa de outra (b), que encontra-se em produção em outro *site*, neste momento a OS (a) atinge o estado de *Espera* - o processo não segue o fluxo enquanto a OS não receber os artefatos gerados pela OS (b). Ao receber os artefatos da OS (b), o processo produtivo volta a seguir o seu fluxo – o estado *Em produção* é novamente caracterizado.

Conforme citado anteriormente, a proposta do DDScrum é alicerçada pelo modelo delineado por L´Erario (2009) e pelo *framework* Scrum. O DSScrum é caracterizado por estados e elementos associados à produção de software (elementos de processo e de produto) – Vide Figura 3.

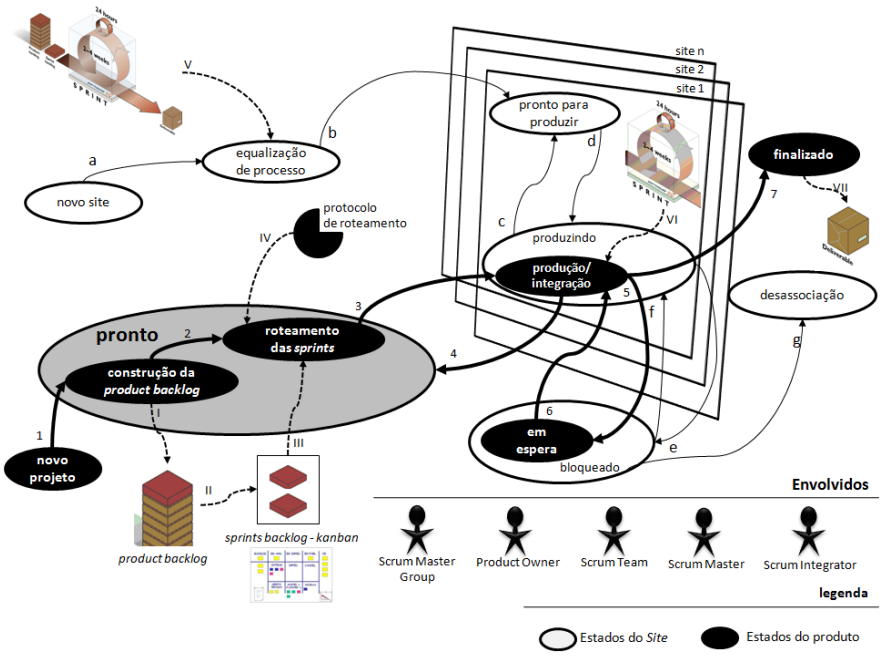

Figura 3 – Proposta do DDScrum

Ao analisar a referida Figura é possível perceber a presença dos estados: *equalizando processo*, *pronto para produzir*, *produzindo*, *bloqueado*, *desassociação*, *pronto* dividido em dois subestados (*construção da product backlog* e *roteamento das sprints*), *produção/integração*, *em*

[2] Uma ordem de serviço reuni um ou mais requisitos de software que serão produzidos por uma determinada entidade.

espera e *finalizado*. Para agilizar a leitura a descrição dos estados, assim como as suas transições foram descritas na Tabela 1.

Tabela 1 – A transição de estados do DDScrum

Estado origem	T[3]	Estado destino	Descrição
novo site	a	equalizando processo	Um novo *site* se candidata a participar da rede de produção.
equalizando processo	b	pronto para produzir	O ***Scrum Master Group*** (***Scrum Master*** de cada *site*) – ***SMG*** - verifica se o *site* candidato a rede de produção possui consistência no processo de produção de software. Celebração de um acordo de produção entre os nós. Após celebração o ***Scrum Master Group*** é acrescido do ***Scrum Master*** do site candidato.
pronto para produzir	d	produzindo	O *site* possui o processo consistente e encontra-se pronto para iniciar a produção das ordens de serviços (OS).
produzindo	c	pronto para produzir	O site recebe uma ordem de serviço e produz artefatos ou subprodutos relacionados ao projeto de software. Finalizada a produção o *site* retorna ao estado anterior.
produzindo	e	bloqueado	O *site* não tem possibilidade de atender mais aos critérios de qualidade (de produção) impostos no acordo celebrado pela rede; não consegue atender demandas tecnológicas ou contratuais (possibilidade da entrega de um determinado produto (ou subproduto) dentro dos prazos e custos pré-estabelecidos) de um novo produto de software ou está aguardando um artefato de produção para continuar o seu trabalho (relação de dependência de artefatos).
bloqueado	f	produzindo	O *site* volta a ter possibilidade de atender aos critérios de qualidade (de produção) impostos no acordo celebrado pela rede; consegue atender demandas tecnológicas ou contratuais (possibilidade da entrega de um determinado produto (ou subproduto) dentro dos prazos e custos pré-estabelecidos) de um novo produto de software ou recebe um artefato de produção para continuar o seu trabalho.
bloqueado	g	desassociação	O *site* permanece por um período longo no estado de bloqueado devido ao não atendimento aos critérios de qualidade impostos pela rede. Este fato leva a *desassociação* do *site* perante a rede de produção. O site pode solicitar a sua dessassociação da rede a qualquer momento.
novo projeto	1	pronto: construção da *product backlog*	Um novo projeto de software é captado pela rede de produção. O ***Scrum Master Group***, constrói o *product backlog.*
pronto: construção da *product backlog*	2	pronto: roteamento das *sprint*	De posse do *product backlog,* o ***Scrum Master Group*** caracteriza as *sprints* – ordens de serviços. Após a caracterização estas *sprints* serão roteadas para os *sites*. Importante: este roteamento segue um protocolo. Esse protocolo deve ser customizado de acordo com as características de cada rede produtiva.
pronto: roteamento das *sprint*	3	produção/integração	As *sprints* são colocadas em produção. A produção e a integração dos componentes de software aglutinados nas *sprints* são de responsabilidade do ***Scrum Team***, ***Scrum Master*** e do ***Scrum Master Group***.
produção/integração	4	pronto	Ao finalizar a produção de uma *sprint* o ***Scrum Team*** e ***Scrum Master*** estão prontos novamente para estabelecer um novo ciclo de produção.
produção/integração	5	em espera	É factível que a produção de uma determinada *sprint* (a) dependa de outra (b), que encontra-se em produção em outro site, neste momento a *sprint* (a) atinge o estado de espera.
em espera	6	produção/integração	Ao receber os artefatos da *sprint* (b), o processo volta a seguir o seu fluxo – o estado em produção é novamente caracterizado – neste momento o ***Scrum Team*** e ***Scrum Integrator*** integram o produto gerado na *sprint* (b) a *sprint* (a).
produção/integração	7	finalizado	Assim que a *sprint* é concluída o estado finalizado é caracterizado. Lembrando que neste momento o ***Product Owner*** recebe um *release* executável do software.

5. Métodos e Procedimentos Aplicados para Validar o Modelo

Para validar o modelo foi aplicado o método de pesquisa experimental. O referido método realiza teste das hipóteses por meio de um experimento controlado, projetado de forma a produzir dados necessários, podendo ser realizado em laboratório ou no próprio campo.

A utilização do método prevê a execução das seguintes atividades: a) Definição da hipótese. b) Concepção do protocolo experimental. Conjunto de regras ambientais e comportamentais na qual se enquadra o experimento. c) Execução do experimento. d) Análise dos resultados (mapeado em uma seção específica devido a sua importância).

a. Definição da hipótese: Para verificar a aplicabilidade do modelo, os autores deste trabalho definiram a seguinte hipótese: **H1**: É possível criar um modelo que alinhe a idéia de desenvolvimento distribuído de software e o *framework* Scrum.

[3] T significa Transição dos estados.

b. Protocolo experimental

Para a concepção do protocolo experimental é necessário:

- Definir o ambiente do experimento. Nesta etapa os pesquisadores devem responder a seguinte questão: O experimento será realizado no laboratório ou no próprio campo do conhecimento?
- Configurar o ambiente: A execução deste passo prevê: 1) Definição das entidades envolvidas no experimento (pessoas, software ou componentes). 2) caracterização das entidades (idade, formação, local de trabalho...); 3) definição da amostra (quantidade de entidades envolvidas no experimento). 4) definição da forma de coleta das informações (aplicação de questionário, observação direta das entidades, avaliação dos resultados gerados segundo um conjunto de critérios.). 5) Avaliação das informações (as informações geradas possuem consistência, são passíveis de generalização?).

As informações inerentes ao protocolo experimental podem ser verificadas no Quadro 1.

Apresentado os métodos e procedimento utilizados neste trabalho, a próxima seção irá descrever a aplicação do DDScrum.

0. Ambiente: Laboratório de Engenharia de Software da Universidade Tecnológica Federal do Paraná – Campus Cornélio Procópio.
1. Entidades: Gerentes de projetos de empresas de desenvolvimento de software que desejam formar uma rede de produção. Estes gerentes foram treinados pelos autores deste trabalho no primeiro semestre de 2013.
2. Caracterização das entidades: Gerentes com formação em Análise e Desenvolvimento de Sistemas, Ciência da Computação e Engenharia da Computação.
3. Definição da amostra: 4 gerentes de 4 empresas – todos com mais de 3 anos de experiência no cargo.
4. Forma de coleta das informações: Observação direta das entidades (durante o treinamento) e depoimento das entidades após a participação do DDScrum. O desenvolvimento do experimento foi realizado em salas separadas. As *sprints* no Scrum variam entre 2 a 4 semanas, no experimento este tempo foi customizado entre 2 a 16 horas.
5. Avaliação: A avaliação será delineada a partir da produção de um software utilizando o DDScrum

Quadro 1 – Protocolo experimental

6. A aplicação do DDScrum

O DDScrum foi aplicado no treinamento, efetuado no mês de maio de 2013, de 4 gerentes de 4 empresas de produção de software. É importante ressaltar que o objetivo destas empresas é compor uma rede de produção.

Durante o treinamento os autores deste trabalho se posicionaram como o *Scrum Master Group – SMG,* apresentaram o DDScrum para os gerentes - estado *equalização do processo* – configuraram dois *sites* de produção, cada um deles com 2 gerentes de projetos – estado *pronto para produzir*. É importante salientar os *sites* foram alocados em salas distintas e comunicação só poderia ser feita por: e-mail, *call-conference* (utilização de um chat) ou telefone celular. Somente os membros do *SMG* tinham acesso direto aos *sites*.

Além de percorrer os dois estados citados no parágrafo anterior, os autores deste trabalho definiram, a partir de um projeto de software (estado *novo projeto*[4]), uma *product backlog*[5] (estado *construção da product backlog*). Além deste artefato os autores do trabalho também definiram o diagrama de entidade e relacionamento (DER[6]) e o modelo de interface[7]. Todos estes artefatos foram publicados nos endereços listados no rodapé desta página.

[4] Vide características em https://dl.dropboxusercontent.com/u/3525445/ddscrum/objetivos.pdf
[5] Vide *product backlog em* https://dl.dropboxusercontent.com/u/3525445/ddscrum/pb.pdf

[6] Vide DER em https://dl.dropboxusercontent.com/u/3525445/ddscrum/der.png

[7] Vide modelo de interface em https://dl.dropboxusercontent.com/u/3525445/ddscrum/interface.pdf

Após a construção dos artefatos, os membros do *Scrum Master Group* se reuniram e iniciaram a configuração das *sprints*, fato este que proporcionou o *roteamento das mesmas*. O *SMG* também configurou a gestão global do projeto utilizando o Kanban. Neste momento os *sites* iniciaram a sua *produção*. Enquanto permaneceram neste estado, os sites mapearam – via kanban: a) o nome da funcionalidade a ser desenvolvida; b) a data início da implementação; c) a data de término; d) a quantidade de horas utilizadas na produção. Ao atingir o estado de espera – fato este ocorrido pelo *site* 2 ao implementar o item 4 da *product backlog* – o *site* apontava quantidade de horas em estado de espera. É possível visualizar o kanban e as informações geradas pelos *sites* na animação apresentada neste *link*: https://dl.dropboxusercontent.com/u/3525445/ddscrum/sprint.ppsx. É importante salientar ainda que os *sites* durante a *produção* do código aplicaram a técnica de desenvolvimento guiado por testes – técnica esta apresentada durante o treinamento.

Ao terminar a implementação os sites notificavam o *SMG* e estes ficaram responsáveis por integrar as funcionalidades geradas – Informações sobre o processo de integração podem ser visualizadas no log de integração (vide: https://dl.dropboxusercontent.com/u/3525445/ddscrum/log.pdf). Os problemas reportados no log eram apresentados aos sites e as manutenções corretivas eram efetuadas, fato este ocorrido com o *site* 2 que ficou responsável pelo desenvolvimento do 4º item da *product backlog* – o site não respeitou as especificações espelhadas no diagrama de entidade e relacionamento[6].

Por fim, ao integrar as funcionalidades agrupadas nas *sprints* o produto (ou parte dele) atinge o estado finalizado e o site encontra-se novamente pronto para produzir.

7. Análise dos Resultados, Conclusões e Perspectivas Futuras

Os resultados obtidos com a aplicação do DDScrum, levando em consideração o objetivo inicial traçado (possibilidade de unir as prerrogativas delineadas pelo desenvolvimento distribuído de software ao *framework* Scrum), pode ser constatado por qualitativamente nos itens abaixo:

- A *equalização do processo* foi feita durante o treinamento, perceba que o DDScrum foi apresentado aos gerentes logo na fase inicial da execução do experimento. Em um ambiente real de produção esta *equalização* pode necessitar de um esforço maior do *SMG*, visto que o *site* candidato a rede já possui um processo de produção, e este deve ser avaliado e posteriormente equalizado. A mudança cultural no processo não é trivial.
- Para efetuar o *roteamento das sprints* o *SMG*, além da *product backlog*, também tiveram que especificar o modelo de dados[6] e o documento de interface[7].
- A tabela de roteamento não foi caracterizada.
- A proposta de uma ferramenta de gestão de projetos alinhada ao kanban, foi gerada, pelos autores deste trabalho, durante o desenvolvimento do experimento – esta ferramenta será apresentada em um trabalho futuro.
- Perceba que o DDScrum foi aplicado a construção e teste do código fonte, as tarefas relacionadas a atividade de projetos de software foram caracterizadas de forma centralizada pelo *SMG*.
- O estado de desassociação não foi atingido durante o experimento.

- Perceba que *site* 2 ficou duas horas no estado de *espera* – valor este capturado pelo *kanban* - https://dl.dropboxusercontent.com/u/3525445/ddscrum/sprint.ppsx.
- Perceba que as *sprints* foram mapeadas na coluna *to do* do Kanban.
- Foi possível verificar que o DDScrum mapeia todos os requisitos de um ambiente de desenvolvimento distribuído: o ambiente distribuído foi constituído por no mínimo dois sites; o experimento proporcionou a distância física entre eles – cada *site* foi alocado em uma sala; uma granularidade de repasse, que identifica a forma na qual o site entrega seu subproduto para a rede foi mapeado; o processo foi difundido; e grau de interação, que revela o quanto a informação é trocada entre os nós, mediante a quantidade de estímulos enviados entre uma instância e outra foi caracterizada durante a integração do produto - participaram da integração *SMG* e o *Scrum Team* de cada *site.*

Dado o contexto delineado para avaliação do modelo, conclui-se que **H1** (é possível criar um modelo que alinhe a idéia de desenvolvimento distribuído de software e o *framework* Scrum) pode ser caracterizada como verdadeira.

Por fim, cabe aos autores deste trabalho proporcionar a implantação do DDScrum nas 4 empresas e comparar os aspectos quantitativos e qualitativos da produção centralizada e distribuída.

8. Referências Bibliográficas

Karolak, D. W. (1998) "Global Software Development – Managing Virtual Teams and Environments". Los Alamitos, EUA. IEEE Computer Society, 159 p.

L'Erario, A. (2009) "M3DS: um modelo de dinâmica de desenvolvimento distribuído de software". 175 p. Tese (Doutorado) – Escola Politécnica da USP. Universidade de São Paulo, São Paulo.

Marquardt, M. J. e Horvath, L. (2001) "Global Teams: how top multinationals span boundaries and cultures with high-speed teamwork". Davies-Black Publishing. Palo Alto, EUA. 246 p.

Prikladnicki, R. e Audy, J. L. N. (2006) "Uma análise comparativa de práticas de Desenvolvimento Distribuído de Software no Brasil e no exterior". In: XX SIMPÓSIO BRASILEIRO DE ENGENHARIA DE SOFTWARE. Florianópolis: SBES. p. 255 – 270

Sutherland, A. *et. al*, (2007) "Distributed SCRUM: Agile Project Management with Outsourced Development Teams" in HICSS'40, Hawaii International Conference on Software Systems, Big Island, Hawaii, 2007.