Um Estudo Exploratório sobre os Efeitos da Refatoração na Coordenação das Atividades de Desenvolvimento de Software Livre

Crys Sayuri Goto, Maryanne Pacheco Rosa, Cleidson R. B. de Souza

Faculdade de Computação – Universidade Federal do Pará

66075 – 110 – Belém – PA – Brasil

{crys.sayuri, rosa.maryanne}@gmail.com, cdesouza@ufpa.br

Resumo. *Este trabalho descreve um estudo cujo objetivo é analisar os efeitos da refatoração na coordenação das atividades de um projeto de software livre. O projeto analisado é o projeto Jackrabbit da Apache Software Foundation. Este projeto foi analisado utilizando-se métricas de análise de redes sociais e testes estatísticos para avaliar hipóteses relacionadas aos possíveis impactos da refatoração na coordenação do projeto. Os resultados iniciais sugerem que o processo de refatoração de fato implica em mudanças na coordenação do projeto, em especial causando uma sobrecarga nos principais engenheiros de software do projeto.*

Abstract. *This paper presents an exploratory study about the effects of refactoring in the coordination of software development activities in open source projects. The project evaluated is the Jackrabbit, an Apache Software Foundation project. This project was analyzed using statistical tests and social networks analysis metrics. We evaluate different hypothesis regarding the possible impacts of the refactoring process on the project coordination. Initial results confirm our hypothesis and suggest that core software developers are especially affected by refactoring activities.*

1. Introdução e Justificativa

Há mais de 30 anos, Parnas (1972) propôs o princípio do ocultamento de informação que deu origem a diversos mecanismos nas linguagens de programação modernas (Larman, 2001). Além das vantagens técnicas, este princípio também tem vantagens gerenciais: quando se divide o software em módulos relativamente independentes, também pode-se dividir a implementação destes módulos em times que podem trabalhar de maneira independente. Em outras palavras, a arquitetura do software[1] e a coordenação das atividades de desenvolvimento estão relacionadas. Diversos estudos quantitativos (Morelli et al., 1995)(Sosa et al., 2002),(Sosa et al., 2003) e qualitativos (Grinter, 1998)(de Souza et al., 2005a) corroboram esta relação sócio-técnica. Por exemplo, MacCormak e colegas (2004) compararam uma abordagem comercial para desenvolvimento de software com outra abordagem de software livre. Visto que os desenvolvedores do projeto comercial encontravam-se todos no mesmo prédio, a arquitetura do software desenvolvido por este time era altamente acoplada. Isto em contraste com o projeto de software livre cujos desenvolvedores eram distribuídos e cujo software final tinha uma arquitetura menos acoplada e mais fácil de manter. Apesar

[1] Neste trabalho, o termo arquitetura de software refere-se à forma como um software é dividido em partes menores e as dependências entre estas partes

do crescente interesse nesta relação sócio-técnica (Cataldo et al., 2008), não existem trabalhos na literatura relacionando-a com o estudo de atividades de refatoração como proposto neste trabalho.

Baseado nesta relação entre arquiteturas de software e a coordenação das atividades de desenvolvimento, este artigo visa avaliar a hipótese de que mudanças na arquitetura do software levam a mudanças na coordenação destas atividades. As modificações na arquitetura serão estudadas a partir de refatorações no código do projeto, enquanto que as mudanças na coordenação das atividades serão estudadas a partir dos padrões de comunicação dos engenheiros de software envolvidos no projeto. Ou seja, este trabalho apresenta um estudo exploratório cujo objetivo é analisar os efeitos da refatoração na coordenação das atividades de um projeto de desenvolvimento de software livre. Ao fazer isto, este artigo também visa identificar potenciais limitações do processo de refatoração.

O restante deste trabalho está organizado da seguinte forma. A seção 2 descreve o projeto Jackrabbit que foi utilizado para investigar a hipótese deste trabalho. A seção 3 apresenta a metodologia utilizada, ou seja, descreve a análise e discussão do estudo de caso deste trabalho e apresenta os resultados de maneira resumida ao final desta seção. A seção 4 apresenta a discussão geral do trabalho, enquanto que a seção 5 descreve as limitações do mesmo. Finalmente, a seção 6 descreve as conclusões e trabalhos futuros.

2. Projeto Jackrabbit

O projeto Apache Jackrabbit é uma implementação completa do Repositório de Conteúdo de Java Technology API (JCR), uma interface padronizada para acessar repositórios de conteúdo. Um repositório de conteúdo é um sistema de gerenciamento de informações que fornece diversos serviços para armazenar, acessar e gerenciar conteúdo. Este projeto foi escolhido por apresentar as características desejadas (Yin, 2003): um conjunto de refatorações no código e, como outros projetos de software livre, todo o seu processo de desenvolvimento disponível na Internet.

O projeto apresenta 4 listas de discussões diferentes,: lista de anúncios (*Jackrabbit Announce List*), lista de usuários (*Jackrabbit Users List*), lista de desenvolvimento (*Jackrabbit Development List*) e lista de controle de código fonte (*Jackrabbit Source Control List*). Contudo, para a realização deste estudo, apenas a lista de desenvolvimento foi analisada, visto que esta lista contém as atividades dos colaboradores do projeto. Para este estudo, usou-se o termo colaboradores para referenciar a todos os envolvidos que participam das listas de discussões do projeto, incluindo aqueles com acesso direto ao código fonte do projeto e que modificam o mesmo e aqueles que contribuem através de submissões de correções e sugestões.

3. Metodologia

3.1 Coleta e Tratamento de Dados

Os dados utilizados neste trabalho foram coletados utilizando o ambiente OSSNetwork (Balieiro et al., 2007). Este ambiente realiza um *parsing* baseado em expressões regulares que faz a leitura do código HTML das páginas web do projeto Jackrabbit e retorna as informações requisitadas do repositório de lista de e-mail. Após obter os dados das páginas web, os mesmos são armazenados em um sistema de gerência de banco de dados. Este banco de dados foi utilizado nesta análise, tal que a partir de

consultas SQL ao mesmo, foi possível extrair informações como: (i) duração de uma *thread*[2] (em dias), (ii) número total de mensagens enviadas em cada *thread*, e (iii) número de colaboradores que enviaram mensagens em cada *thread*[3]. Estes dados foram analisados de duas formas diferentes. Primeiro, através de testes estatísticos não-paramétricos (Wild e Seber, 1999). E segundo, usando a técnica de análise de redes sociais (Wasserman e Faust, 1994). As redes sociais são geradas a partir das informações provenientes das listas de e-mail referentes à lista de desenvolvimento do projeto, tendo como nós as pessoas que postaram mensagens nas listas e as arestas correspondem ao fato de uma pessoa ter respondido a mensagem da outra. Assim, em uma determinada *thread*, se um ator A posta uma mensagem e um ator B responde essa mensagem, então é criada uma aresta de B para A. Da mesma maneira se um ator C responde a mensagem do ator B, é criada uma aresta de C para B, e assim sucessivamente. A análise de uma rede social requer que a rede a ser analisada comporte-se como um grafo conectado (Wasserman e Faust, 1994), ou seja, para a realização da análise das redes sociais do projeto Jackrabbit foi necessário retirar os componentes e atores isolados que as redes apresentavam.

3.2 Identificação das Refatorações

A identificação dos períodos de refatorações do projeto Jackrabbit foi possível com o auxílio da ferramenta JIRA (uma ferramenta para a gerência de bugs) e através de leituras feitas nas listas de discussões do projeto. Diversas refatorações foram encontradas desde o início do projeto até a data deste trabalho. Cada refatoração corresponde a uma *Issue* específica no sistema JIRA e, portanto, possui uma página HTML própria com informações como: a data de criação e de encerramento da *Issue*, arquivos afetados pela *Issue* e assim por diante. A data de encerramento da refatoração depende de uma mudança de status da *Issue* correspondente, ou seja, depende de um participante do projeto modificar o status da *Issue* para *closed*. Observou-se que a modificação de status de algumas *Issues* ocorre após longos períodos (até mesmo anos) depois da abertura da mesma, mesmo que nenhuma atividade associada a esta *Issue* seja registrada. Isto sugere que a data de modificação do status da *Issue* não reflete o término real de uma refatoração. Desta forma, definiu-se que o período considerado para uma determinada refatoração iniciaria no dia que a *Issue* que a descreve foi criada e terminaria no dia da última mensagem que descrevesse uma efetiva modificação realizada no código do projeto.

3.3 Divisão dos períodos de análise

Como o objetivo deste trabalho é estudar o efeito da refatoração em um projeto de software livre, a idéia principal consiste em fazer uma comparação entre os períodos "antes de" e "durante" cada uma das refatorações. Entretanto, algumas refatorações tem duração de apenas 1 dia. Desta forma, 3 refatorações diferentes foram analisadas, mas agregadas em apenas dois períodos de análise diferentes:

- Refatoração 1 (R1) – realizada no período de 03 de março de 2005 a 16 de março de 2005. Corresponde a 2 refatorações diferentes: JCR-53 e JCR-66 da ferramenta JIRA; e

[2] Uma *thread* corresponde a uma discussão sobre um assunto específico, ou ainda, uma discussão onde o Assunto (Subject) da mensagem permanece o mesmo.
[3] Deve-se observar que neste trabalho o conteúdo de cada mensagem da t*hread* não foi avaliado, apenas o número de mensagens existente na mesma.

- Refatoração 2 (R2) – realizada no período de 26 de janeiro de 2006 a 07 de março de 2006 e corresponde a *Issue* JCR-309 da ferramenta JIRA.

Os períodos para a análise foram divididos da seguinte maneira:

- Período *antes* da refatoração (*An*) –período correspondente a 1 mês antes do início da refatoração *n*; e
- Período *durante* a refatoração (*Rn*) –período correspondente exatamente ao período que a refatoração *n* aconteceu, considerando-se as datas discutidas anteriormente.

Para cada um dos períodos citados foram calculados os valores da média e do desvio padrão para os dados de duração de uma *thread*, número de mensagens de uma *thread* e número de colaboradores que enviam mensagens em uma *thread*.

3.4 Testes Realizados

A primeira análise realizada foi baseada em testes estatísticos. Inicialmente, o teste de Kolmogorov-Smirnov (Wild and Seber, 1999) foi utilizado para determinar se os dados seguem ou não uma distribuição normal. Para todos os períodos analisados, os dados coletados não apresentavam distribuição normal. Logo, testes não-paramétricos (Wild and Seber, 1999) foram escolhidos visando verificar se os dados eram "equivalentes", ou seja, se existia uma diferença *significativa* entre os dados do período antes da refatoração (A*n*) e os dados do período *durante* a mesma (R*n*).

No caso dos dados de duração das *threads*, número de mensagens por *thread* e número de colaboradores por *thread*, o teste Mann-Whitney (Wild and Seber, 1999) foi utilizado para fazer a comparação entre os dados de A*n* e R*n*. Este teste visa comparar a equivalência entre duas amostras independentes, ou seja, os dados de A*n* correspondem, por exemplo, aos tempos de duração das *threads* abertas durante o período A*n* enquanto que os dados de R*n* correspondem aos tempos de duração das *threads* abertas durante o período R*n*. Por se tratarem de *threads* diferentes para cada período, os dados são ditos independentes entre si. Desta forma, o teste Mann-Whitney verifica se a média do tempo de duração das *threads* nos períodos A*n* e R*n* são ou não equivalentes.

A segunda análise visava comparar o número de mensagens enviadas por cada colaborador nos períodos *antecedentes* as refatorações (A*n*) e o número de mensagens enviadas pelos mesmos colaboradores *durante* os períodos de refatorações (R*n*). A comparação entre os dados de A*n* e R*n* foi feita utilizando-se o teste Wilcoxon, um teste não-paramétrico utilizado para comparar grupos de dados pareados (Wild and Seber, 1999). Neste caso, os dados são ditos pareados porque eles consistiam no número de mensagens que *um mesmo colaborador* enviou antes e durante as refatorações.

Finalmente, o número de mensagens enviadas por cada colaborador pertencente ao *core*[4] das redes sociais do projeto Jackrabbit foi calculado para os mesmos períodos A*n* e R*n*. Estes dados foram comparados utilizando novamente o teste Wilcoxon, visto que também não seguiam uma distribuição normal. Neste caso, a comparação foi feita somente com os colaboradores que enviaram mensagens nos períodos A*n* e R*n e* que pertenciam ao *core* das redes sociais geradas em cada um destes períodos. O core das redes sociais foi calculado utilizando-se a ferramenta UCINET[5]. Esta ferramenta

[4] O *core* de uma rede social é um conjunto de nós centrais que possuem densas relações entre si e com os nós mais periféricos (Borgatti, 2000). O conceito de *core* consiste em uma extensão do conceito de centralização (Wasserman e Faust, 1994).

[5] http://www.analytictech.com/ucinet/ucinet.htm

permite a análise de redes sociais através de diversas rotinas de análise, além de um módulo integrado para desenhar gráficos de redes sociais. .

3.5 Resultados

Os resultados obtidos para duração, número de mensagens e número de colaboradores por *thread* (Tabela 1) indicam que não existem diferenças significativas quando os períodos A1 e R1 são comparados, pois o valor de significância (*p-value*) obtido para as refatorações A1 e R1 nas três situações apresentadas são maiores que 0,05 (nível de significância adotado). Já para as refatorações A2 e R2, o valor de significância é menor que 0,05. Estes resultados indicam que, por exemplo, a duração média das *threads, o* no número de mensagens por *thread* e o número de colaboradores por *thread* antes e durante a refatoração não possui diferença significativa para os períodos A1 e R1. Contudo, para os períodos A2 e R2 o resultado indica que houve uma diminuição significativa na duração, no número de mensagens e no número de colaboradores por *thread* durante a refatoração, quando comparado com o período anterior a mesma.

Tabela 1. Resumo dos testes para *Threads*

- Duração A1_R1 (p=0,647) → duração A1 = duração R1;
- Duração A2_R2 (p=0,004) → duração A2 > duração R2;

- Mensagem A1_R1 (p=0,786) → nº msg A1 = nº msg R1;
- Mensagem A2_R2 (p=0,016) → n° msg A2 > n° msg R2;

- Colaborador A1_R1 (p=0,636) → n° colab A1 = n° colab R1;
- Colaborador A2_R2 (p=0,006) → n° colab A2 > n° colab R2;

Em relação ao número de mensagens enviadas por cada colaborador (Tabela 2) os resultados sugerem que houve um aumento significativo no envio de mensagens no período das refatorações. Da mesma forma, os resultados dos testes quando se utilizam apenas os dados dos colaboradores presentes no *core* das redes sociais (Tabela 3) sugere que houve um aumento significativo no envio de mensagens no período das refatorações. Em outras palavras, os colaboradores que faziam parte do *core* da rede social antes e durante o período de refatoração enviaram um número maior de mensagens durante a refatoração quando comparado ao período anterior a refatoração.

Tabela 2. Resumo dos testes para número de mensagens enviadas por colaborador

- NumMsg A1_R1 (p=0,055) → n° msg A1 < n° msg R1
- NumMsg A2_R2 (p=0,0001) → n° msg A2 < n° msg R2

Tabela 3. Resumo dos testes para número de mensagens enviadas por colaborador no core

- NumMsg A1_R1 (p=0,002) → n° msg A1 < n° msg R1
- NumMsg A2_R2 (p=0,004) → n° msg A2 < n° msg R2

4. Discussão

As médias do tempo de duração, do número de mensagens e do número de colaboradores durante o período de refatoração permanecem equivalentes às médias do período anterior à refatoração nas refatorações R1. Em outras palavras, as discussões

(representadas pelas *threads*) na lista de e-mails do projeto continuaram a ocorrer da mesma forma que ocorriam antes. Entretanto, isto não é verdade para o período de refatoração R2: existe *forte evidência* ($p < 0.01$) indicando que a média da duração, número de mensagens e número de colaboradores envolvidos nas *threads* no período da segunda refatoração (R2) é *menor* que no período que antecede a refatoração (A2).

Em outras palavras, R1 não parece causar nenhum efeito nas discussões que ocorrem entre os desenvolvedores. Entretanto, isto não é verdadeiro para R2. Buscou-se então uma comparação entre estas 2 refatorações para se entender a diferença no resultado. Ao se analisar o objetivo da refatoração R2 conforme descrito na ferramenta JIRA e apresentado na Tabela 4, é possível observar que a refatoração R2 correspondeu à modificação e criação de APIs para um novo pacote. Isto é, a refatoração R2 modifica múltiplos módulos enquanto que as duas refatorações englobadas no período R1 modificam apenas um único módulo (Carver, 2007).

Tabela 4. Descrição da Refatoração R2

Refatoração R2 (JCR-309) *JCR – 309* *Refatoração para criação de interface* Período: 26/01/06 - 07/03/06 • Melhorou a documentação e rastreou as extensões do JCR (Repositório de Conteúdo de Java Technology API); • Rastreou os componentes API fornecidos pelo projeto Jackrabbit; e Permitiu mais espaço para a refatoração no núcleo do projeto.

Em resumo, uma possível explicação para o resultado distinto para R1 e R2, é que a refatoração R2 é uma refatoração que afeta de maneira mais significativa o projeto, pois modifica APIs do mesmo, em contraste com a refatoração R1 que gera modificações isoladas. Assim, conjectura-se que os colaboradores do projeto tenham diminuído suas atividades ao perceberem o início da refatoração para evitar que a mesma tivesse algum impacto em suas atividades.

Os resultados da análise da comparação do número de mensagens enviadas por cada colaborador no período anterior à refatoração e no período da refatoração sugerem que cada desenvolvedor envia um número maior de mensagens durante as refatorações. Isto é, a média do número de mensagens enviadas por desenvolvedor durante as refatorações (R*n*) é maior que a mesma média durante os períodos anteriores às refatorações (A*n*). Este aumento no número de mensagens enviadas, apesar de significativo, pode não causar influência nas atividades de coordenação do projeto. Por isso, decidiu-se avaliar apenas o *core* do projeto. Isto é, o teste estatístico foi repetido, mas apenas para os colaboradores presentes no *core* do projeto. Os resultados também indicam um aumento significativo no número de mensagens que eles enviaram, ou seja, durante os períodos de refatoração, os colaboradores pertencentes ao *core* enviaram, em média, um número maior de mensagens do que no período que antecede a refatoração. Este resultado é particularmente importante por dois motivos. Primeiro, porque ele é independente da refatoração analisada, isto é, ele ocorre durante R1 e R2. E segundo, porque os membros do *core* são, por definição, os colaboradores mais ativos no projeto, aqueles que mais participam das discussões. Estes participantes tem papel primordial na continuidade do projeto, visto que eles detém conhecimento sobre o mesmo e mantém o mesmo ativo ao darem continuidade às discussões (*threads*). De um modo geral, os membros do core do projeto são aqueles que o criaram ou que o gerenciam (de Souza et

al. 2005b). O fato de que estes colaboradores precisam “trabalhar mais” durante os períodos de refatoração sugere que as refatorações devem ser planejadas cuidadosamente para evitar distúrbios às atividades de desenvolvimento de software.

5. Limitações do Trabalho

O trabalho desenvolvido apresenta algumas limitações, as quais deverão ser sanadas em trabalhos futuros. Primeiro, o projeto Jackrabbit foi o único projeto analisado para o estudo em questão, desta forma, os resultados deste projeto não podem ser generalizados para outros projetos de software livre. Segundo, este projeto apresenta diversas listas de discussão, contudo apenas a lista de discussão dos desenvolvedores foi levada em consideração. Neste trabalho assumiu-se, com um alto grau de confidência, que esta lista retratava a comunicação entre os desenvolvedores do projeto, podendo assim ser utilizada para avaliar os efeitos que cada refatoração ocasionava na coordenação das atividades dos participantes do projeto. Entretanto, é possível que as outras listas de discussão também contenham informações relativas à coordenação das atividades do projeto. Em terceiro lugar, o projeto Jackrabbit apresenta oito refatorações, porém para a realização deste estudo foram selecionadas apenas duas. A análise do efeito destas refatorações foi realizada de forma quantitativa, ou seja, o estudo teve como base o número de mensagens enviadas por cada colaborador, mas não executou análise do código modificado pelas refatorações. Finalmente, o estudo restringiu-se a avaliar apenas os períodos de duração das refatorações e os períodos anteriores as refatorações, mas seria interessante fazer uma análise englobando todos os períodos existentes no projeto, ou seja, um estudo envolvendo os períodos anteriores a cada refatoração, os períodos durante cada refatoração e também os períodos *posteriores* às refatorações.

6. Conclusões e Trabalhos Futuros

Este trabalho teve como objetivo estudar a relação existente entre a estrutura de um projeto de desenvolvimento de software e a coordenação das atividades do mesmo. Baseado nesta relação, a hipótese aqui investigada é que o processo de refatoração, ao implicar em modificações na arquitetura de software de um projeto, também implicaria em mudanças na coordenação das atividades do mesmo. Desta forma, um estudo exploratório foi realizado com o objetivo de investigar possíveis mudanças na coordenação das atividades durante os períodos que ocorreram refatorações no projeto Jackrabbit. As avaliações foram feitas para os períodos que antecederam as refatorações (*An*) e os períodos durante a realização das refatorações (*Rn*).

Este trabalho utilizou testes estatísticos não-paramétricos e análise de redes sociais para estudar os efeitos na coordenação das atividades. Esta coordenação foi estudada a partir da análise da lista de discussão do projeto e considerando-se informações (tempo de duração, número de mensagens e número de colaboradores) das *threads* desta lista e do número de mensagens enviadas por colaborador (utilizando todos os colaboradores e apenas os colaboradores que pertencem ao *core* de A*n* e ao *core* de R*n*). Os resultados dos testes sugerem as *threads* da lista discussão do projeto podem diminuir em duração, numero de mensagens e participantes dependendo do tipo de refatoração em andamento. Além disso, os colaboradores do projeto, durante as refatorações, enviam um maior número de mensagens em comparação ao período que antecede às refatorações. Isto é verdade para os principais participantes deste projeto, aqueles que fazem parte do *core* da rede social formada a partir das discussões entre os mesmos. Com base

nesses resultados, é possível afirmar que a hipótese testada nesse trabalho realmente é verdadeira, ou seja, o processo de refatoração implicou em mudanças na coordenação das atividades do projeto Jackrabbit. Deve-se ressaltar que estes resultados são limitados, dado o contexto no qual este trabalho foi realizado.

Pretende-se dar prosseguimento a este trabalho, considerando-se todos os períodos de interesse: os períodos anteriores a cada refatoração, os períodos durante cada refatoração e, finalmente, os períodos posteriores às refatorações. Além disso, novos projetos de software livre estão sendo considerados para análise, assim como análise multi-variada (Wild e Seber, 1999) para eliminar outros fatores (número de colaboradores, tamanho da refatoração, etc) que possam influenciar os resultados.

Agradecimentos

Este trabalho foi financiado pelo CNPq através do Edital Universal 20006 (projeto 479206/2006-6) e pela Universidade Federal do Pará.

7. Referências Bibliográficas

Balieiro, M. A.; Souza Jr., S. F. d. S.; et al. OSSNetwork: Um Ambiente para Estudo de Comunidades Virtuais de Software Livre usando Redes Sociais. IV Experimental Software Engineering Latin America Workshop, São Paulo, SP, 2007.

Borgatti, S. Models of Core-Periphery Structures. Journal of Social network analysis. Vol. 21, pp. 375-395, 2000.

Carver, J. Comunicação pessoal, Setembro de 2007.

Cataldo, M. et al. Workshop on Socio-Technical Congruence, International Conference on Software Engineering, Leipzig, Alemanha, 10 de Maio de 2008.

de Souza, C. R. B. On the Relationship between Software Dependencies and Coordination: Field Studies and Tool Support. Donald Bren School of Information and Computer Sciences. Irvine, CA, University of California, Irvine. Ph.D.: 186, 2005a.

de Souza, C.R.B., Froehlich, J. and Dourish, P., Seeking the Source: Software Source Code as a Social and Technical Artifact. in ACM Conference on Supporting Group Work, (Sanibel Island, FL, USA), ACM Press, 197-206, 2005b.

Grinter, R.E., Recomposition: Putting It All Back Together Again. in Conference on Computer Supported Cooperative Work, (Seattle, WA, USA, 1998), 393-402.

Larman, G. Protected Variation: The Importance of Being Closed. IEEE Software, 18 (3). 2001.

MacCormack, A., et al. (2004). Exploring the Structure of Complex Software Designs: An Empirical Study of Open Source and Proprietary Code. Harvard Business School Working Papers. Cambridge, MA, Harvard University**:** 40.

Morelli, M. D.; Eppinger, S. D.; et al. Predicting Technical Communication in Product Development Organizations. IEEE Trans. on Engineering Management 42(3): 215-222.

Parnas, D. L. On the Criteria to be used in Decomposing systems into modules. 1972.

Sosa, M. E.; et al. Factors that influence Technical Communication in Distributed Product Development: An Empirical Study in the Telecommunications Industry. IEEE Transactions on Engineering Management 49(1): 45-58, 2002.

Sosa, M. E.; Eppinger, S. D.; et al. Identifying Modular and Integrative Systems and Their Impact on Design Team Interactions. ASME Journal of Mech. Design 125: 240-252, 2003.

Yin, R. (2003). "Case Study Research, Design and Methods, 3rd edition, Sage Publications.

Wasserman, S. and Faust, K. Social Network Analysis: Methods and Applications. Cambridge University Press, Cambridge, UK, 1994.

Wild, C. J.; Seber, G. A. F. Chance Encounters: A First Course in Data Analysis and Inference, John Wiley & Sons, 1999.