UNIVERSIDADE FEDERAL DO PARÁ
CENTRO DE CIÊNCIAS EXATAS E NATURAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MAYARA COSTA FIGUEIREDO

UM ESTUDO EMPÍRICO SOBRE ARQUITETOS DE TECNOLOGIA DA INFORMAÇÃO: IMPLICAÇÕES PARA A PESQUISA EM ARQUITETURA DE SOFTWARE

DM 05/2011

UFPA / ICEN / PPGCC
Campus Universitário do Guamá
Belém-Pará-Brasil
2011

UNIVERSIDADE FEDERAL DO PARÁ
CENTRO DE CIÊNCIAS EXATAS E NATURAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MAYARA COSTA FIGUEIREDO

UM ESTUDO EMPÍRICO SOBRE ARQUITETOS DE TECNOLOGIA DA INFORMAÇÃO: IMPLICAÇÕES PARA A PESQUISA EM ARQUITETURA DE SOFTWARE

Dissertação submetida à Banca Examinadora do Programa de Pós-Graduação em Ciência da Computação da UFPA para a obtenção do Grau de Mestre em Ciência da Computação

UFPA / ICEN / PPGCC
Campus Universitário do Guamá
Belém-Pará-Brasil
2011

Figueiredo, Mayara Costa

Um estudo empírico sobre Arquitetos de Tecnologia da Informação: Implicações para a pesquisa em arquitetura de software / (Mayara Costa Figueiredo); orientador, Cleidson Ronald Botelho de Souza. ï 2011.

172f. il. 28cm.

Dissertação (Mestrado) - Universidade Federal do Pará. Instituto de Ciências Exatas e Naturais. Programa de Pós-Graduação em Ciência da Computação. Belém, 2011.

1. Engenharia de Software. 2. Arquitetura de Software I. Souza, Cleidson Ronald Botelho de, orient. II. Universidade Federal do Pará, Instituto de Ciências Exatas e Naturais, Programa de Pós-Graduação em Ciência da Computação. III. Título.

CDD 22. ed. 005.1

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MAYARA COSTA FIGUEIREDO

UM ESTUDO EMPÍRICO SOBRE ARQUITETOS DE TECNOLOGIA DE INFORMAÇÃO: IMPLICAÇÕES PARA A PESQUISA EM ARQUITETURA DE SOFTWARE

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal do Pará como requisito para obtenção do título de Mestre em Ciência da Computação, defendida e aprovada em 26/04/2011, pela banca examinadora constituída pelos seguintes membros:

Prof. Dr. Cleidson Ronald Botelho de Souza
Orientador – PPGCC/UFPA

Prof. Dr. Rafael Prikladnicki
Membro Externo – PUC/RS

Profª. Drª. Tayana Uchôa Conte
Membro Externo – UFAM/AM

Visto:
Prof. Dr. Sandro Ronaldo Bezerra Oliveira
Coordenador do PPGCC/ICEN/UFPA

Prof. Dr. Sandro Ronaldo Oliveira
Coordenador do PPGCC/UFPA
Mat 1643578

Dedico este trabalho a meus pais.

AGRADECIMENTOS

Agradeço Deus, a minha família e a todos os envolvidos no meu processo de formação, todos que me incentivaram e me apoiaram sempre que precisei. Agradeço ao meu pai e à minha mãe, as pessoas mais importantes da minha vida, que sempre me deram amor e bons exemplos, sempre ao meu lado nos momentos bons e ruins. Ao meu irmão por estar sempre comigo me ajudando em tudo que pode. Ao meu orientador, Prof. Dr. Cleidson R.B. de Souza pela orientação desde a graduação, pela paciência e principalmente confiança. Ao Prof. Dr. Rafael Prikladnicki, por ter nos indicado os contatos das empresas e apoiado o desenvolvimento do trabalho em Porto Alegre. A Marcelo Zílio, pelo trabalho em conjunto nas primeiras etapas de pesquisa. A todos os informantes por terem dedicado tempo e atenção para o trabalho. Aos membros do LINC (Laboratório de pesquisa em Interação e Colaboração) pela companhia e apoio de sempre, e aos meus amigos que são muito importantes e significam muito na minha vida.

A todos vocês, o meu mais sincero Obrigada!

Mayara Figueiredo.

"Ciência da Computação tem tanto a ver com
o computador como a Astronomia com o
telescópio, a Biologia com o microscópio,
ou a Química com os tubos de ensaio.
A ciência não estuda ferramentas, mas o que
fazemos e o que descobrimos com elas."
(citação atribuída a Edsger Dijkstra)

SUMÁRIO

LISTA DE FIGURAS

RESUMO

Na academia, a disciplina de arquitetura de software geralmente é estudada focando em aspectos como estilos arquiteturais, padrões de projeto, *frameworks* e outros também ligados à engenharia de software, ou seja, nas tecnologias e processos necessários para criá-la. Entretanto, aspectos como clientes, o negócio da organização, a infraestrutura de Tecnologia da Informação (TI) existente e outros também influenciam na construção de um sistema de software. Assim, é importante que a arquitetura de software seja estudada dentro do contexto maior no qual ela é desenvolvida, a arquitetura de TI. Apesar disso, existem poucos estudos que focam no trabalho diário dos arquitetos de software, suas práticas de trabalho, ferramentas e necessidades, e um número ainda menor de estudos que consideram o trabalho dos arquitetos de software no contexto da arquitetura de sistemas de Tecnologia da Informação. Considerando esta lacuna, este trabalho descreve um estudo qualitativo conduzido com arquitetos de TI. Foram realizadas 27 entrevistas semiestruturadas com 21 informantes diferentes em 9 diferentes empresas desenvolvedoras de sistemas. A análise dos dados foi feita usando técnicas da Teoria Fundamentada em Dados. Os resultados sugerem que existem diferentes tipos de arquitetos (incluindo o arquiteto de software) em todas as empresas pesquisadas. Esses tipos de arquitetos, que podem ser institucionalizados ou não, dividem as atividades de arquitetura entre si, uma vez que possuem atividades diferentes, interagem com *stakeholders* diferentes e contribuem de formas diferentes para as soluções arquiteturais das empresas. Além disso, as atividades realizadas por eles estão interconectadas de tal forma que a arquitetura de software não pode ser estudada de maneira isolada, pois a sua criação é largamente influenciada pelo cenário mais amplo da arquitetura de TI da empresa. Apesar disso, observando as ferramentas, *frameworks* e linguagens de modelagem que oferecem apoio às atividades de arquitetura constatou-se que, embora existam mecanismos voltados especificamente para os arquitetos de TI, estes geralmente não apoiam o trabalho desempenhado nos vários papéis ou não focam nas interconexões entre estes papéis. Estas interconexões são importantes pois as atividades dos arquitetos de TI (o que inclui os arquitetos de software) são divididas, mas não são desacopladas e é através dessas interconexões que eles trocam e repassam conhecimento, contribuindo para a difusão de informações e compartilhamento de conhecimento nos projetos. Finalmente, a partir dos resultados apresentados nesta dissertação, possíveis implicações para a pesquisa em arquitetura de software são discutidas, especialmente aquelas relacionadas a mecanismos de apoio às atividades dos arquitetos de TI e difusão de informações e compartilhamento de conhecimento.

Palavras-chave: arquitetura de TI, arquitetura de software, papéis de arquiteto, compartilhamento de conhecimento, estudo qualitativo, teoria fundamentada em dados.

ABSTRACT

Software architecture is often studied focusing on technical aspects like architectural styles, patterns, frameworks, and other important items related to software engineering, i.e. in the technologies and processes used to design a software architecture. However, additional aspects such as customers, organizationôs business, existing Information Technology (IT) infrastructure, and others also influence the construction of a software system. So, it is important to study software architecture considering this broader context in which it is embedded, an organizationôs IT architecture. Despite that, there are few studies that focus in the daily work of software architects, their work practices, tools and needs, and an even smaller number that takes into account the work of software architects in the broader context of Information Technology system architecture. Considering this gap, this work describes a qualitative study conducted with IT architects. Therefore, 27 semi-structured interviews with 21 different interviewees of 9 different system-developing companies were conducted and analyzed using grounded theory methods. The results suggest that there are different architecture roles (including software architects) in every researched company. These architect roles, which can be institutionalized or not, divide software architecture activities among themselves, once they have different activities, interact with different stakeholders and contribute in different ways to the companies architectural solutions. Furthermore, the activities performed by them are highly interconnected in such a way that the software architecture can not be studied in an isolated way, because its design is highly influenced by the IT architecture broader scenario. Despite that, observing the tools, frameworks and modeling languages that offer support to architecture activities it was found that, although there are mechanisms oriented specifically to IT architects, they usually do not support all architecture roles together or have focus in their interconnections. These interconnections are important because the IT architects activities (which include software architects) are divided but not decoupled and it is through these interconnections that they exchange and pass knowledge, contributing to information and knowledge diffusion in the projects. Finally, based on the results present in this dissertation, some possible implications to the software architecture research are discussed, specially the ones related to support mechanisms to IT architects activities and information diffusion and knowledge sharing.

Keywords: IT architecture, software architecture, architect roles, knowledge sharing, qualitative study, grounded theory.

CAPÍTULO 1

1. Introdução

1.1. Visão geral

É fato que a arquitetura de software é um dos ñcomponentesò principais de um sistema de software (Taylor *et al.*, 2010) e, por isso, demanda bastante atenção e estudo. Taylor *et al.* (2010) definem a arquitetura de software como o conjunto das principais decisões de *design* tomadas durante o desenvolvimento de sistemas e qualquer evolução do mesmo. Entretanto, Smolander (2002) afirma que existe uma diferença considerável entre a percepção da arquitetura de software da academia e da indústria. A primeira foca em definições explícitas da arquitetura, usando linguagens e ferramentas descritivas, e abordando-a sob a perspectiva de uma engenharia de software tecnicamente avançada. Por outro lado, a segunda utiliza principalmente descrições informais que são determinadas pela necessidade da situação, e foca na necessidade de a arquitetura de software ser comunicada e compreendida por um conjunto diversificado de *stakeholders* (Smolander, 2002). Além disso, com o advento da globalização, as empresas de Tecnologia da Informação (TI) começaram a modificar seu foco para organizações de soluções, passando assim a demandar dos profissionais conhecimentos tanto da empresa quanto das soluções de TI (Morneau e Talley, 2007).

Dessa forma, as demandas da indústria acrescentam à arquitetura de software a necessidade de lidar com aspectos organizacionais (conhecimento sobre a empresa) e colaborativos (comunicação) que influenciam as decisões de *design* do sistema de software produzido. Quando se desenvolve uma arquitetura de software na indústria é preciso considerar aspectos mais organizacionais tanto da empresa que desenvolve a arquitetura de software quanto do próprio cliente, uma vez que o principal objetivo é atender à uma necessidade de um cliente através do desenvolvimento de sistemas computacionais. Dessa forma, as soluções arquiteturais na indústria incluem sistemas, aplicações e componentes de processos, podendo envolver também a aplicação e integração de uma variedade de produtos, tecnologias e serviços, várias arquiteturas de sistemas e aplicações, passando por componentes tanto de hardware quanto de software. Essas soluções são documentadas e chamadas de arquiteturas de TI (The Open Group, 2008), da qual a arquitetura de software é

apenas uma parte. Portanto, é preciso que a academia enxergue o contexto maior onde a arquitetura de software está inserida na indústria: a criação de soluções para os problemas de negócio dos clientes através da aplicação de tecnologia da informação.

1.2. Motivação

Na academia, em cursos de Ciência da Computação, é comum existirem disciplinas que enfoquem o estudo da arquitetura de software, nas quais comumente se ensinam estilos arquiteturais, a importância de requisitos não funcionais, padrões de projetos relacionados, linguagens de descrição de arquitetura, etc (Clements *et al.*, 2007)(Shaw *et al.*, 2005)(Shaw e Van Vilet, 2005). No currículo de referência da Sociedade Brasileira de Computação (SBC)para cursos de graduação em computação e informática (SBC, 1999) são descritas as disciplinas indicadas tanto para cursos de ciência da computação quanto sistemas de informação. No primeiro, não se observa nenhuma disciplina que tenha ênfase em arquitetura, nem de software nem de TI. O que se deduz é que tópicos sobre arquitetura de software estão incluídos dentro de outras disciplinas, especialmente na engenharia de software, cujos tópicos são os seguintes: processo de desenvolvimento de software, ciclo de vida de desenvolvimento de software, qualidade de software, técnicas de planejamento e gerenciamento de software, gerenciamento de configuração de software, engenharia de requisitos, métodos de análise e de projeto de software, garantia de qualidade de software, verificação, validação e teste, manutenção, documentação, padrões de desenvolvimento, reuso, engenharia reversa, reengenharia e ambientes de desenvolvimento de software. Dessa forma, deduz-se que aspectos de arquitetura de software são ministrados dentro dos tópicos desta disciplina, uma vez que não se observa um tópico claro sobre o assunto. Já na parte do documento referente a sistemas de informação é possível encontrar na seção sobre as competências de um profissional de sistemas de informação o seguinte item: ñconceber e especificar a arquitetura de tecnologia da informação capaz de suportar os sistemas de informações das organizaçõesò. Entretanto, apesar de arquitetura de tecnologia da informação ser citada, o que se encontra nas disciplinas é apenas um tópico sobre arquitetura de software dentro da disciplina chamada de interface homem-máquina.

Como se pode observar, o currículo de referência para cursos de graduação em computação costuma apresentar apenas aspectos relativos à arquitetura de software, por vezes sem mesmo definir um tópico específico para o assunto. Entretanto, como descrito na seção

anterior, a arquitetura de software é apenas parte de um conceito maior de arquitetura de TI, portanto, o que é ensinado na academia compreende apenas uma parte pequena, ainda que importante, do que arquitetura significa para a indústria. Além disso, ainda há a diferença existente entre a percepção da própria arquitetura que academia e indústria possuem, como indicado por Smolander (2002) e mencionado na seção anterior. Esse cenário dificulta o treinamento de profissionais nesta área, os chamados arquitetos, uma vez que as atividades nas quais eles são treinados durante o tempo de academia não englobam aspectos considerados importantes para a indústria (Shaw *et al.*, 2005)(Shaw e Van Vilet, 2005).

Quando se observa as atividades de arquitetura no âmbito da indústria, é comum encontrar vários cargos e papéis que levam o título de arquiteto no nome e que possuem atividades muito diferentes uns dos outros, tais como arquiteto de sistemas, arquiteto de infraestrutura e arquiteto de aplicações, configurando assim um título que varia muito de empresa para empresa. Como se pode perceber, as atividades realizadas por arquitetos, especialmente no contexto de arquitetura de TI, não são padronizadas, portanto, apesar da importância de suas atividades, o papel do arquiteto ainda é muito ambíguo (Akenine, 2008)(Clements *et al.*, 2007)(Hofstader, 2008)(Kralj, 2008)(Unde, 2008). Assim, o primeiro obstáculo para alguém que almeja se tornar um arquiteto é definir no que exatamente ele precisa focar e adquirir treinamento, devido a esta ambiguidade e ao fato de os valores que o arquiteto provê não serem facilmente quantificados (Hofstader, 2008)(Unde, 2008).

Por outro lado, apesar dessa ambiguidade, o papel de arquiteto vem sendo cada vez mais reconhecido pela indústria. Recentemente, uma pesquisa do Payscale e CNNMoney (CNNMoney, 2010), dois sites voltados para o mercado de trabalho nos EUA, identificou os 100 melhores empregos nos Estados Unidos, considerando salário e perspectiva de crescimento. O emprego de arquiteto de software, um dos tipos de arquitetos encontrados na indústria, aparece como número um, ou seja, como o melhor trabalho na atualidade nos Estados Unidos e ainda com boas perspectivas de crescimento (CNNMoney, 2010). Essa classificação reflete a importância que o papel de arquiteto vem adquirindo na atualidade e motiva ainda mais estudos relacionados às atividades que este papel desempenha, uma vez que a partir do momento que for mais bem definido esse conjunto de atividades básicas será mais fácil formar profissionais para exercer esse papel. Além disso, compreender as atividades que os arquitetos realizam no dia a dia pode contribuir para os próprios arquitetos

identificarem possíveis focos de estudo ou especialização e a necessidade de ferramentas de apoio.

Assim, este trabalho pretende focar neste papel, o de arquiteto, tão importante e ao mesmo tempo tão ambíguo, especialmente considerando o contexto de TI onde um arquiteto trabalha na indústria. Pretende-se identificar as implicações que este contexto e a forma como a indústria organiza suas atividades de arquitetura trazem para a pesquisa e ensino de arquitetura de software. A partir do momento em que se entendem tais implicações, é possível fazer sugestões que estejam alinhadas com as necessidades do mercado para a formação destes profissionais, podendo, em longo prazo, contribuir também com a indústria.

1.3. Justificativa

Taylor *et al*. (2010) afirmam que todo e qualquer sistema possui uma arquitetura de software e que esta não deve ser encarada como uma fase de desenvolvimento de um projeto, suas atividades devem permear o projeto todo. Eles também afirmam que toda aplicação tem pelo menos um arquiteto (podendo ser também um grupo de arquitetos), que pode não ter este título ou mesmo não ser reconhecido como tal. Sendo assim, essa pessoa (ou grupo) é quem tomará as principais decisões de *design* do projeto e deverá estar envolvido nele como um todo, uma vez que as atividades de arquitetura, como afirmado por Taylor *et al*. (2010), devem estar presentes em todas as fases do projeto. De fato, Hofstader (2008) afirma que um arquiteto de software efetivo deve estar envolvido no contínuo de todas as etapas de um projeto de desenvolvimento de software. Além disso, Bass *et al*. (2003) afirmam que a arquitetura de software é um conjunto de ideias técnicas e de negócio, portanto aspectos relativos à organização e ao tipo de negócio em que ela é produzida devem ser considerados. Na verdade essa relação entre a organização e a arquitetura já é apontada desde 1968, quando Conway criou o que ficou conhecido como a ñLei de Conwayò, que diz que qualquer organização que produza um sistema de software irá produzir um *design* cuja arquitetura será um espelho da estrutura de comunicação da organização (Conway, 1968). Com isso, devem-se concentrar os esforços de arquitetura não apenas nos aspectos técnicos da estrutura do software que o projeto produz (ou seja, na arquitetura de software), mas também nos aspectos relativos à própria organização, isto é, os aspectos não-técnicos. Considerando que o arquiteto é o principal papel encarregado das atividades de arquitetura, ele deverá saber avaliar tais aspectos e criar o *design* da arquitetura considerando ambos. Nesta mesma linha, Ovaska *et al*. (2003) afirmam que a arquitetura de software deve ser usada como um mecanismo de

coordenação, mas a utilidade deste uso vai depender de pessoas chave. O papel de arquiteto de software é um dos ñpapéis chaveò para isso. Ele deve trabalhar de forma a favorecer que todos os envolvidos no projeto tenham um entendimento comum sobre o mesmo (Grinter, 1999)(Ovaska *et al.*, 2003), algo muito importante para uma coordenação efetiva do projeto. Assim, o papel de arquiteto de software, através de suas atividades de arquitetura, também é responsável por apoiar a coordenação de um projeto. Isso aumenta ainda mais a sua importância, uma vez que problemas de coordenação são uma das principais causas de atraso e estouro de orçamento (Ovaska *et al.*, 2003).

Apesar da importância do papel do arquiteto de software e do conhecimento da relação entre aspectos organizacionais e arquitetura de software existir desde 1968, a maioria dos estudos enfoca simplesmente nos aspectos técnicos (Smolander, 2002)(Shaw *et al.*, 2005)(Shaw e Van Vilet, 2005). Além disso, existem poucos estudos que focam no trabalho dos arquitetos de software. Em um deles, Grinter (1999) estudou o trabalho que estes realizam para coordenar o *design* através das fronteiras organizacionais e institucionais. Neste, ela identificou vários aspectos não técnicos relacionados ao trabalho dos arquitetos de software, como por exemplo a necessidade de eles trabalharem como *boundary spanners*[1]dentro dos projetos, para assim contribuir para a criação e manutenção do conhecimento comum que é extremamente útil para a coordenação. Além disso, Unphon e Dittrich (2010) identificaram em seu trabalho que é muito mais comum que o conhecimento da arquitetura de software fique armazenado na cabeça dos arquitetos do que documentado e que os *stakeholders* preferem consultar o arquiteto a pesquisar em um documento, aumentando ainda mais a importância deste papel.

Como se pode perceber, as atividades de arquitetura envolvem muito mais aspectos do que apenas definir a estrutura técnica do software que será produzido em um dado projeto. Elas envolvem, além dos aspectos técnicos, os aspectos organizacionais da empresa, como o tipo de negócio, de clientes, infraestrutura, número de projetos; e aspectos sociais, como coordenação e difusão de informações (Grinter, 1999)(Ovaska *et al.*, 2003). Portanto, como afirmado por Taylor *et al*. (2010), a arquitetura não é apenas uma fase de desenvolvimento, suas atividades estão presentes no projeto como um todo, assim como o papel de arquiteto também deve estar.

[1] Um *boundary spanner* interage com vários tipos de *stakeholders* e através dessas interações difundem informações no projeto e ajudam na coordenação do mesmo (Grinter, 1999)(Ovaska *et al.*, 2003)

Dessa forma, torna-se importante entender como as pessoas que ocupam este papel realizam suas atividades e mesmo quais são essas atividades, pois apenas desta forma é possível compreender melhor o processo real de desenvolvimento de software, e assim, criar formas de apoiar o trabalho dessas pessoas. Além disso, entendendo como a indústria realiza e coordena seu desenvolvimento de software, especialmente no que diz respeito à arquitetura, pode contribuir para a formação de profissionais que estejam mais alinhados com as necessidades da indústria.

1.4. Questão de Pesquisa

Como descrito nas seções anteriores, ao se projetar uma arquitetura de software se está lidando com vários assuntos que não se limitam aos aspectos técnicos, modelo arquitetural ou padrões de projeto devem ser adotados. É preciso conciliar esses itens a aspectos relativos à organização, negócio e, ao mesmo tempo, ser capaz de auxiliar na coordenação das atividades de desenvolvimento direta ou indiretamente. Com isso, o papel de arquiteto adquire novas perspectivas que não são apenas técnicas, e, tais perspectivas, variam em cada empresa transformando-o em um papel ambíguo (Akenine, 2008)(Hofstader, 2008)(Unde, 2008). Entretanto, apesar da importância desse papel e dessa dificuldade de padronização de suas atividades o que se percebe é que este é um papel pouco estudado. De fato, a maioria dos estudos sobre arquitetura trata de aspectos técnicos da própria arquitetura. Smolander (2002) descreveu em seu estudo quatro metáforas através das quais se costuma enxergar a arquitetura e constatou que a maioria dos estudos que são realizados são relativos à parte técnica da arquitetura de software. Clements (2007) afirma que arquitetura como uma profissão (incluindo seus deveres, habilidades e conhecimento) é uma área que não é exaustivamente estudada; ou seja, a maioria dos estudos trata da arquitetura e não das pessoas que realizam as atividades de arquitetura, os arquitetos.

A partir dessas constatações tem-se o seguinte quadro: a maioria dos estudos enfatiza a parte técnica da arquitetura de software (Smolander, 2002) e justamente os papéis que desempenham as atividades de arquitetura (técnicas ou não), os arquitetos, são pouco estudados pela academia e ambíguos na indústria devido à falta de padronização dessas atividades (Unde, 2008). Em resumo, constata-se que poucos estudos pesquisam o trabalho dos arquitetos na prática. Com isso, esta pesquisa foca no trabalho das pessoas que realizam as atividades de arquitetura, visando compreender as nuances de suas atividades, uma vez que são elas que darão significado ao domínio, ou seja, é a partir das atividades destas pessoas, e

da interpretação que elas dão para estas atividades, que se torna possível compreender o fenômeno estudado. Para isso, o trabalho começa com uma questão de pesquisa aberta para poder adquirir um conhecimento amplo sobre as atividades e refinar esse conhecimento conforme a pesquisa evolui. Assim, a questão de pesquisa é: ñComo os arquitetos de TI realizam suas atividades de arquitetura na indústria?ò Ao responder esta pergunta espera-se capturar e compreender quais são as atividades de arquitetura, quem as realiza e como é a interação entre essas pessoas.

1.5. Objetivos

Esta dissertação tem como objetivo principal identificar e analisar as atividades que os arquitetos realizam na prática. Assim, pretende-se compreender como os arquitetos de TI desempenham suas tarefas no dia a dia de suas empresas, de forma a identificar quais aspectos influenciam no trabalho destes profissionais, especialmente aqueles não técnicos ligados à organização e aspectos sociais.

Como objetivo específico deste trabalho, pretende-se analisar como as atividades que os arquitetos realizam na indústria influenciam no ensino e pesquisa em arquitetura de software, como por exemplo, identificar aspectos normalmente não abordados em pesquisas acadêmicas, especialmente aqueles relacionados a aspectos organizacionais e sociais. Com isso, pretende-se indicar estes aspectos para academia, para que assim esta possa formar profissionais melhor preparados para atender às necessidades da indústria. Com isso, pretende-se contribuir para uma aproximação entre academia e indústria. Em outras palavras, os objetivos específicos deste trabalho são:

- Realizar um estudo sobre aspectos organizacionais e sociais que influenciam na arquitetura;
- Identificar as atividades que os arquitetos de TI realizam na indústria, especialmente aquelas relacionadas a aspectos não técnicos;
- Identificar como a arquitetura de TI influencia na arquitetura de software;
- Identificar como o trabalho dos arquitetos de TI na indústria pode influenciar a pesquisa e o ensino em arquitetura de software; e
- Investigar como o arquiteto de TI participa e qual a sua influência sobre a coordenação e colaboração dentro dos projetos.

1.6. Contribuições esperadas

Baseado no contexto descrito anteriormente, pretende-se identificar as atividades que os arquitetos realizam na indústria, especialmente aquelas relacionadas a aspectos não técnicos (organizacionais e sociais) e assim contribuir com um estudo sobre tais aspectos de forma a ressaltar a sua importância e contribuição para o bom andamento de projetos reais. Para isso serão investigados quais aspectos sociais e organizacionais são mais exigidos dos arquitetos e como eles e as empresas lidam com estes aspectos.

Alguns estudos já apontam que é necessário incluir na formação dos profissionais esses aspectos não técnicos (Berenbach, 2008)(Morneau e Talley, 2007)(Shaw *et al.*, 2005)(Shaw e Van Vilet, 2005). Assim, conduzindo um estudo que identifique as principais atividades dos arquitetos de TI na indústria pode ser possível gerar sugestões que possam ser adotadas pela academia, inclusive na formação destes profissionais. Com isso, em longo prazo este tipo de estudo pode inclusive contribuir para a indústria, caso sugestões baseadas em informações da indústria sejam adotadas na formação dos profissionais de arquitetura.

Portanto, pretende-se com este trabalho contribuir para a academia, através da compreensão do trabalho real realizado pelos arquitetos, e das consequentes implicações desta compreensão no desenvolvimento de ferramentas, metodologias ou na própria pesquisa e ensino de arquitetura de software.

1.7. Organização do Trabalho

Esta dissertação está organizada em seis capítulos e quatro apêndices organizados da seguinte forma:

- CAPÍTULO 1 ï descreve a motivação, objetivos e contribuições, como também a organização do documento da dissertação;
- CAPÍTULO 2 ï apresenta a revisão de literatura através da comparação de diversos trabalhos relacionados com o tema da dissertação;
- CAPÍTULO 3 ï apresenta a metodologia adotada neste estudo, descrevendo os principais conceitos relacionados e como o pesquisa foi conduzida;
- CAPÍTULO 4 ï apresenta os resultados encontrados durante a pesquisa;
- CAPÍTULO 5 ï apresenta a discussão sobre os resultados expostos no capítulo anterior;

- CAPÍTULO 6 ï apresenta as conclusões deste trabalho, suas contribuições e limitações e os trabalhos futuros que podem ser realizados;
- REFERÊNCIAS ï contém uma lista das referências utilizadas na elaboração deste trabalho;
- APÊNDICE A ï apresenta o questionário de caracterização das empresas;
- APÊNDICE B ï apresenta o guia usado na primeira etapa de entrevistas;
- APÊNDICE C ï apresenta o guia usado na segunda etapa de entrevistas;e
- APÊNDICE D ï apresenta o guia usado na terceira e quarta etapas de entrevistas.

CAPÍTULO 2

2. A Arquitetura de TI e os Arquitetos

2.1. Visão geral

Como citado no capítulo anterior, o currículo de referência da SBC (SBC, 1999) para ciência da computação foca na construção de software, isto é, os conceitos, as ferramentas, tecnologias e os processos utilizados para se desenvolver um software. Nesse sentido, um dos conceitos estudados é o de arquitetura de software que Taylor *et al*. (2010) definem como o conjunto das principais decisões de *design* tomadas durante o desenvolvimento do sistema e qualquer evolução do mesmo. Nesse âmbito, costuma-se focar o estudo da arquitetura de software em aspectos como padrões e modelos arquiteturais, *frameworks*, etapas de processo, entre outros. Entretanto, quando se observa o processo de construção de um sistema de computação na indústria é preciso considerar também a estrutura da organização como um todo: seu negócio, sua infraestrutura incluindo os outros sistemas que ela já utiliza, seu pessoal, seus clientes etc. E, nesse sentido, a arquitetura de um sistema computacional torna-se um conceito mais amplo do que simplesmente arquitetura de software, ou seja, a estrutura e as decisões de *design* do software que se desenvolve em um determinado projeto. Ela engloba atividades e aspectos além daqueles relativos à arquitetura de software especificamente, sendo chamada de arquitetura de TI, da qual a arquitetura de software é apenas uma parte. Entretanto, apesar desse cenário, não é possível identificar no currículo de referência de ciência da computação tópicos que correspondam a estes aspectos. Já no currículo de sistemas de informação observa-se que, apesar de este currículo citar que os profissionais devem ter a competência de conceber e especificar a arquitetura de TI, há apenas um tópico dentro das disciplinas voltado ao estudo de arquitetura de software (e não de TI como era de se esperar) (SBC, 1999).

Apesar disso, um arquiteto não pode se limitar a conhecimentos técnicos relativos à arquitetura de software, uma vez que, na indústria, este profissional terá que lidar com diversos desafios que exigirão muito mais que o conhecimento de *frameworks*, modelos arquiteturais e padrões de projeto. Entretanto, o que se percebe hoje é que não existe um padrão definido para as atividades que um arquiteto deve executar (Unde, 2008), uma vez que isso varia de organização para organização. Além disso, o arquiteto, não importando o nome

do cargo que ocupa, interage com uma grande diversidade de *stakeholders*, e cada grupo de *stakeholders* espera que ele haja de uma forma diferente, tornando este papel ainda mais ambíguo e complexo (Unde, 2008). Assim, além das atividades que um arquiteto deve desenvolver não serem padronizadas, ele ainda deve interagir com diferentes *stakeholders* que esperam que ele desempenhe atividades diferentes.

No restante deste capítulo, primeiramente aspectos sobre arquitetura de TI serão descritos, de forma a apresentar as atividades que ela engloba. Depois o papel dos arquitetos de TI será discutido, descrevendo os aspectos técnicos e não técnicos relacionados. Por fim, alguns aspectos de arquitetura de software e de colaboração e coordenação serão discutidos devido ao fato de tais aspectos serem muito importantes para o trabalho dos arquitetos.

2.2. Arquitetura de TI

Segundo Morneau e Talley (2007) o termo arquitetura de software começou a aparecer na literatura a partir dos trabalhos de Djikstra e Parnas, sendo que o primeiro, em 1968, indicou a importância de se preocupar com a forma com a qual o software é estruturado ao invés de apenas programá-lo, enquanto o segundo, em 1972, discutiu sobre a importância da decomposição de um sistema em módulos. Ainda segundo Morneau e Talley (2007), a arquitetura se tornou mais presente nos anos 90, quando a IEEE padronizou a descrição arquitetural no padrão 1471-2000.

Hoje, ao se falar de arquitetura de software sob a ótica da indústria é preciso considerar muito mais do que os requisitos não funcionais, os modelos arquiteturais, padrões de projetos, entre outros fatores, de um determinado software que a empresa está construindo. É preciso que a organização tenha uma estratégia e um processo de governança tecnológicos que sejam intrinsecamente relacionados com as diretrizes de negócio da empresa, apoiando e influenciando as mesmas, mas também sendo influenciados por ela (The Open Group, 2004). Dessa forma, não é possível concentrar a definição de arquitetura nos aspectos relativos à arquitetura de software, uma vez que estes vão ser apenas uma parte da arquitetura de TI da empresa.

A IASA[2], uma associação global de arquitetos de TI (IASA, 2011), tomou a iniciativa de definir de forma precisa e estruturada o que é arquitetura de TI (Wilt, 2010). Para isso a IASA realizou primeiramente um estudo qualitativo usando grupos focais com arquitetos da

[2]*International Association of Software Architects*

indústria buscando saber qual a definição deles para a arquitetura de TI. Depois disso, a IASA buscou expandir os conceitos encontrados através de uma pesquisa quantitativa, na qual enviaram um *survey* a aproximadamente 7.000 arquitetos.

Com estes resultados, a IASA redigiu a seguinte definição para arquitetura de TI: ña arte ou ciência de projetar e fornecer estratégias valiosas de negócioò (Wilt, 2010). Essa definição mostra que arquitetura de TI não é voltada para a entrega de soluções ou projetos; ela trata de entregar estratégias tecnológicas, das quais as soluções e projetos são apenas uma parte.

A IASA usou como inspiração o modelo de outras profissões, como a medicina, onde os médicos passam por processos de especializações, mas antes todos recebem uma educação generalizada. Com isso, a IASA então pretende identificar o conjunto de atividades comuns que diferenciam a profissão de arquiteto de TI das demais profissões, uma vez que o título geral da profissão deve ser significativo antes que as especializações o sejam (Preiss, 2008)(Wilt, 2010).

A partir deste estudo com grupos focais e questionário realizado pela IASA e da norma IEEE 1471, a própria IASA criou o ITABoK, ou corpo de conhecimento de arquitetura de TI (do inglês *IT Architecture Body of Knowledge*) (Wilt, 2010). Nesse documento, ela identificou os cinco pilares da arquitetura de TI: estratégia tecnológica de negócio, dinâmica humana, atributos de qualidade, *design* e ambiente de TI (Wilt, 2010). Tais pilares constituem o que a IASA chama de fundamentação da arquitetura de TI, isto é, as habilidades fundamentais que qualquer arquiteto, não importa a especificidade do seu trabalho ou especialização que tenha, deve possuir. Dessa forma, esses cinco pilares constituem o conhecimento geral que todos os arquitetos devem ter e acima disso estão as especializações, como esquematizado na Figura 2.1.

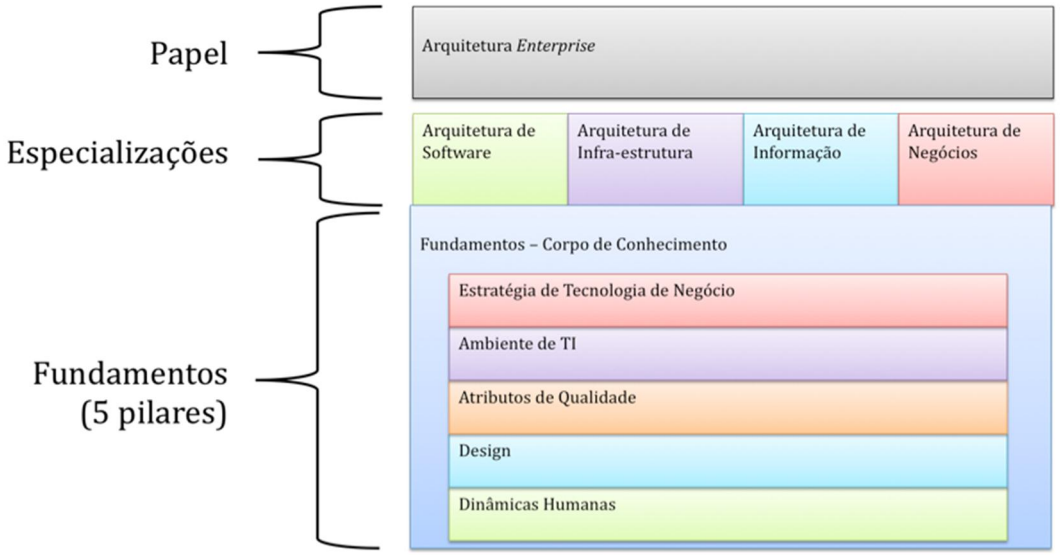

Figura 2.1 Matriz de habilidades da IASA (Wilt, 2010)

A seguir, uma breve descrição dos cinco pilares que compreendem as habilidades básicas que um arquiteto de TI deve possuir (Wilt, 2010):

- Estratégia tecnológica de negócio: este pilar não está necessariamente relacionado com TI em si, ele está mais relacionado com habilidades de negócio. Isto inclui entender como a organização se enquadra no ambiente competitivo do seu negócio de atuação, trabalhar com inovação e tendências. Também está muito relacionado não apenas em conhecer os cálculos financeiros, mas também em saber usá-los para justificar suas decisões arquiteturais, especialmente ao negociar com CIOs e CEOs.

- Ambiente de TI: está relacionado com a organização da tecnologia da empresa de forma a verificar a maturidade da solução e da própria empresa. Abrange aspectos de desenvolvimento de aplicações, gerência de mudanças, governança, infraestrutura, gerência técnica de projeto, plataformas e *frameworks*, gerência de ativos e métodos técnicas e ferramentas de testes.

- Atributos de qualidade: são características não funcionais de um componente ou sistema que permeiam todo o desenvolvimento. Tais atributos, que envolvem usabilidade, manutenibilidade, flexibilidade entre outros, muitas vezes são conflitantes e é necessário balanceá-los e definir prioridades entre eles.

- *Design*: as habilidades deste pilar são a ferramenta primária para construir uma estratégia de arquitetura para o negócio. Deve-se considerar e documentar as justificativas, razões e análises de conflitos, uma vez que todas as decisões, artefatos e

elementos de *design* devem estar ligados a um requisito de negócio. Esse pilar implica em conhecer metodologias e técnicas de *design*, ferramentas para *design* e *design* de artefatos, como padrões, estilos e visões. É necessário compreender o sistema como um todo, incluindo interconexões com outros sistemas, o ambiente completo onde ele existe e as perspectivas da indústria, tanto relacionadas a padrões internos quanto externos.

- Dinâmicas humanas: este pilar está associado a gerenciar as pessoas e suas inter-relações no contexto do projeto ou ambiente de TI. Envolve políticas internas, gerência de diferentes culturas, relacionamento com o cliente, *mentoring*, negociação e colaboração, liderança, habilidades de apresentação e escrita, entre outros.

Essas seriam as habilidades básicas nas quais todo arquiteto deveria receber treinamento, segundo a IASA. Por sua vez, a organização The Open Group (The Open Group, 2010) em sua certificação para arquitetos de TI (ITAC ï *IT Architect Certification*) afirma que existem disciplinas de arquitetura diferentes, como arquitetura de negócios e arquitetura de software (The Open Group, 2008). A IASA considera essas disciplinas especializações da arquitetura de TI. Os conceitos dos dois grupos estão alinhados, uma vez que para a ITAC (certificação do The Open Group) é esperado que os arquitetos adquiram proficiência, ou se especializem, em uma das disciplinas.

A IASA definiu as especializações de forma bem genérica, para não se prenderem a produtos de mercado. Com isso, ela identificou 4 especializações principais: arquitetura de software, arquitetura de infraestrutura, arquitetura de informação e arquitetura de negócio.

A primeira, a arquitetura de software, consiste do conjunto de estruturas necessárias para pensar na construção do sistema, compreendendo elementos de software, os relacionamentos entre eles, as propriedades de ambos e toda a documentação relacionada (IASA, 2010). Por sua vez, a arquitetura de infraestrutura está relacionada com a descrição da estrutura e do comportamento da infraestrutura de tecnologia da organização, da solução ou do sistema, abrangendo a configuração de hardware, aplicações de infraestrutura que são usadas, a infraestrutura de serviços oferecida para as aplicações e os protocolos e redes que as conectam, tanto do cliente quanto do servidor. Tudo isso considerando aspectos relativos a desempenho, resiliência, armazenamento e backup (IASA, 2010).

A arquitetura de informação trata do modelo ou conceito da informação usada em atividades que requerem detalhes explícitos de sistemas complexos. Esta arquitetura compreende atividades de sistemas de bibliotecas, sistemas de gerência de conteúdo, desenvolvimento web, interações de usuários, desenvolvimento de bancos de dados, programação, escrita técnica, arquitetura *enterprise* e *design* de sistemas de software críticos (IASA, 2010). Por fim, a última especialização é a de arquitetura de negócio. Esta especialização está relacionada com a organização arquitetural do negócio e os documentos e diagramas que a descrevem (IASA, 2010).

É importante explicar a ñcamadaò superior da Figura 2.1, chamada *Enterprise Architecture,* que na verdade está desconectada do resto. Para IASA a arquitetura *enterprise* não é uma especialização, e sim um papel. Eles a definem como uma descrição rigorosa da estrutura de uma empresa, o que inclui componentes *enterprise* (entidades de negócio), as propriedades externamente visíveis dos componentes e os relacionamentos entre eles, descrevendo os princípios guia para os requisitos, *design* e evolução da empresa. Tal descrição é abrangente e inclui objetivos organizacionais, processos de negócio, papéis, estruturas e comportamentos organizacionais, informações de negócio, aplicações de software e sistemas de computação (IASA, 2010). Ela está propositalmente desconectada do todo na Figura 2.1 porque o arquiteto *enterprise* não gerencia nem comanda os outros arquitetos e algumas empresas possuem inclusive um arquiteto *enterprise* para cada linha de negócio da organização (Wilt, 2010). Na próxima seção são apresentadas mais informações sobre o arquiteto *enterprise* e as outras especializações de arquitetos.

2.3. O papel do arquiteto de TI

Como mencionado anteriormente, as empresas de TI vêm mudando seu foco, demandando dos profissionais, além dos conhecimentos técnicos, conhecimentos da própria empresa e de seus negócios. Atualmente, com a difusão e avanço da tecnologia, os empregos baseados nas habilidades puramente técnicas acabaram se tornando *commodities* que podem ser distribuídas para outras localidades que ofereçam o mesmo trabalho por um custo menor (Morneau e Talley, 2007). Com isso, especialmente nos países desenvolvidos, as necessidades procuradas nos trabalhadores da área de TI, em geral, passam a incluir, além da experiência em tecnologia e gerência de projetos, outras atividades como liderança e habilidade de seguir um

líder[3], comunicação, colaboração, resolução de problemas, pensamento crítico, criatividade, inovação, experiência em negócios globais e autogerenciamento (Morneau e Talley, 2007).

Ao mesmo tempo, com o crescimento das iniciativas de projetos distribuídos geograficamente, cresce também a necessidade de as linhas gerais da solução arquitetural estarem definidas antes de começar a construção (Morneau e Talley, 2007), pois a solução arquitetural influencia diretamente como a coordenação dos projetos distribuídos irá ocorrer (Ovaska *et al.*, 2003). Entretanto, apesar da importância de as linhas gerais das arquiteturas serem concebidas cedo o suficiente no projeto, especialmente quando se trabalha com Desenvolvimento Distribuído de Software (DDS), não se deve fazer da arquitetura uma fase de projeto ñestáticaò: as atividades de arquitetura, mesmo tendo uma fase definida para começarem, devem existir durante o projeto todo (Taylor *et al.*, 2010). Com isso, os arquitetos devem estar envolvidos em todas as etapas de um projeto de desenvolvimento de software (Hofstader, 2008).

Um arquiteto de TI define as soluções para os problemas de negócio do seu cliente através da aplicação de tecnologia da informação. Tais soluções incluem sistemas, aplicações e componentes de processos, podendo envolver também aplicação e integração de uma variedade de produtos, tecnologias e serviços, várias arquiteturas de sistemas e aplicações, passando por componentes tanto de hardware quanto de software. Como mencionado no Capítulo 2, essas soluções são documentadas e chamadas de arquiteturas (The Open Group, 2008).

As necessidades de as linhas gerais da arquitetura serem pensadas cedo o suficiente (especialmente em projetos distribuídos) e de o arquiteto acompanhar todas as etapas do projeto tornam as atividades deste papel ainda mais importantes. Por isso, é interessante desenvolver estratégias para apoiar e melhorar tais atividades. Entretanto, para poder indicar aos arquitetos como eles podem melhorar seu trabalho é necessário antes entender quais são as atividades que eles realmente realizam (Clements *et al.*, 2007). Essa importância é ressaltada devido ao *gap* que existe entre academia e indústria (Smolander, 2002).

As atividades técnicas, tais como *design*, avaliação tecnológica e documentação são bem conhecidas (Clements *et al.*, 2007) e são mais frequentemente abordadas pela academia (Smolander, 2002), mas não representam o todo. Os arquitetos passam a maior parte do tempo

[3] do inglês *Followership*

em atividades que não são especificamente de produzir uma arquitetura para o projeto em que trabalham, logo atividades não técnicas (Clements *et al.*, 2007). Existem muitas atividades relacionadas à interação dos arquitetos com diferentes *stakeholders*, além de pesquisa, revisões, entre outras. Dessa forma, o papel de arquiteto de TI compreende tanto atividades técnicas quanto não técnicas, como será abordado a seguir.

2.3.1. Aspectos técnicos

Os aspectos técnicos são os mais comumente associados ao trabalho dos arquitetos e estão relacionados com a criação da arquitetura em si, sendo que os estudos geralmente focam na arquitetura de software. Hofstader (2008) usou a sua definição de arquitetos de TI para tratar dos aspectos técnicos deste papel. Para ele ño papel de arquiteto é resolver um problema pela definição de sistemas através da aplicação de conhecimento abstrato e métodos comprovados para um conjunto de tecnologias com o objetivo de criar uma solução extensível e de fácil manutençãoò. A partir desta definição, ele fez as seguintes considerações: para resolver um problema o arquiteto precisa conhecer muito bem o domínio do problema, para definir um sistema usando tecnologia ele precisa ter experiência técnica, para aplicar conhecimento abstrato ele precisa ser capaz de conceitualizar a solução técnica, e, finalmente, para usar de métodos comprovados o arquiteto precisa antes conhecer e entender os padrões usados para resolver problemas similares. Dessa forma, o arquiteto precisa ser capaz de aplicar essas habilidades chave na definição e desenvolvimento de uma solução de software e de TI (Hofstader, 2008).

Ao lembrar os cinco pilares descritos por Wilt (2010) que compreendem a base do conhecimento do papel de arquiteto de TI definido pela IASA, percebe-se que 3 deles são mais voltados para os aspectos técnicos: atributos de qualidade, *design* e ambiente de TI. O primeiro, como explicado na seção 2.2 corresponde às características não funcionais de um componente ou sistema no qual o arquiteto trabalhe. Estas características são normalmente as primeiras a serem cortadas quando as pressões de tempo e orçamento aumentam, porém também são as primeiras a apresentar problemas quando o sistema é lançado. Já o pilar *design* é considerado a principal ferramenta para que o arquiteto exerça seu trabalho, pois, de uma forma superficial, é fazendo uso das habilidades relacionadas a este pilar que o arquiteto desenvolverá a estratégia tecnológica da qual é encarregado. Por fim, o pilar ambiente de TI está relacionado com a infraestrutura tecnológica da organização, incluindo plataformas, *frameworks*, gerência de ativos, testes etc.

Hofstader (2008) divide o conjunto de habilidades de um arquiteto de TI em dois espaços: espaço do problema e espaço da solução. O primeiro está relacionado com definir o escopo do problema e da solução que será desenvolvida e, por isso, requer que o arquiteto conheça e compreenda o que será construído, tenha conhecimento do domínio e a conceitualização de como a informação fluirá através da solução. Por sua vez, o segundo espaço está mais relacionado com o desenvolvimento da solução em si, uma vez que o arquiteto precisa compreender que esta não consiste apenas na implementação dos requisitos funcionais usando plataformas de desenvolvimento, *frameworks* e tecnologias de infraestrutura, mas também está relacionada com os requisitos não funcionais da solução que, se forem negligenciados, aumentam a probabilidade do sistema falhar (Hofstader, 2008).

A seguir serão enumeradas as atividades técnicas mais comumente relacionadas ao papel de arquiteto de TI:

1. Conhecimento de domínio:

   É muito importante que o arquiteto compreenda o problema antes de definir uma solução para ele. Isso inclui reconhecer e estar ciente de requisitos que não foram claramente definidos pelo cliente ou que são requisitos que não dependem dele, como requisitos legais e de regulamentações (Unde, 2008). Segundo Hofstader (2008), o domínio do problema pode ser horizontal ou vertical. O horizontal é aplicável através de diferentes indústrias, enquanto o vertical é específico de uma indústria em particular. Ambos possuem padrões organizacionais que devem ser observados ao se desenvolver a estratégia tecnológica da empresa (Hofstader, 2008). Além disso, muitas organizações produzem muitos sistemas dentro do mesmo domínio, o que pode facilitar o trabalho do arquiteto, pois existem arquiteturas já específicas para estes domínios. Entretanto, há empresas que não possuem uma linha de desenvolvimento de sistemas pertencente a um mesmo domínio e, dessa forma, o arquiteto precisa compreender muito bem o domínio de cada projeto que ele participará, suas propriedades e idiossincrasias (Taylor *et al.*, 2010).

2. *Design*:

   Como a IASA cita no seu pilar correspondente, esta é a ferramenta primária do arquiteto de TI (Wilt, 2010). Esta habilidade envolve decisões, artefatos e elementos de *design* criados para a estratégia tecnológica que o arquiteto deve criar.

Esta atividade está também relacionada com a habilidade do arquiteto de lidar com complexidades. Existem dois tipos de complexidades: a fundamental e a acidental. A *fundamental* é algo intrínseco do problema de negócio que o software vai apoiar. Tais complexidades algumas vezes podem ser simplificadas, mas em outras o arquiteto terá que lidar com elas. Na verdade, cabe ao arquiteto tanto identificar quando é possível simplificar, quanto lidar com a complexidade quando não o é. É necessário para isso que o arquiteto seja habilidoso para que ele possa compreender e modelar o problema corretamente (Akenine, 2008). Por sua vez, *a complexidade acidental* surge a partir das escolhas tecnológicas e arquiteturais feitas durante o desenvolvimento da solução. Decisões relativas a atributos de qualidade, tais quais gerenciabilidade, escalabilidade, segurança e reusabilidade, devem ser tomadas com a participação de diferentes *stakeholders*, cujas necessidades podem estar em desacordo. Essas decisões causam as complexidades acidentais e um dos papéis mais importantes do arquiteto é saber balanceá-las de forma efetiva (Akenine, 2008).

Assim, o arquiteto precisa ser capaz de reconhecer, reutilizar ou inventar soluções de *design* que sejam efetivas e também aplicá-las de forma adequada. Reciprocamente, ele precisa reconhecer *designs* não efetivos, estilos e/ou padrões inapropriados e decisões de *design* com consequências indesejadas (Taylor *et al.*, 2010). Isso envolve documentar as justificativas, razões e considerações de conflitos tomadas por ele, bem como ter conhecimento de metodologias, técnicas e ferramentas de *design*, padrões, estilos, visões, *viewpoints*, *frameworks*, entre outros, sempre buscando visualizar o sistema como um todo.

3. Tecnologista:

Para um arquiteto é essencial ter sido exposto a uma amplitude de tecnologias e fornecedores de plataformas para compreender seus pontos fracos e fortes e, assim, fazer a melhor escolha para os requisitos do seu problema (Unde, 2008). Produzir um produto que obedeça a todos os padrões de qualidade, como performance e escalabilidade, requer um bom conhecimento técnico. Entretanto, compreender como a tecnologia funciona não é suficiente para desenvolver uma solução de software robusta. É essencial compreender onde a tecnologia é aplicável no contexto de uma solução para assim desenvolver um produto de qualidade (Hofstader, 2008). Dessa forma, o arquiteto precisa compreender como a tecnologia será aplicada na sua

solução, pois um *design* ótimo é inútil se não puder ser implementado (Taylor *et al.*, 2010).

É impossível entender cada detalhe de todas as tecnologias, porém um arquiteto deve compreender no mínimo o objetivo e aplicabilidade de qualquer tecnologia antes de requerer o seu uso no projeto. Além disso, o arquiteto deve mapear a tecnologia nos requisitos funcionais ou na parte conceitual da arquitetura a que se refere, sempre explicando o objetivo da utilização daquela tecnologia (Hofstader, 2008).

Um outro ponto importante referente à escolha das tecnologias: não deixar que a paixão por novas tecnologias ou novidades de mercado se sobressaírem ao problema que elas devem resolver. É necessário focar nos requisitos do problema (funcionais e não funcionais) e encontrar a quantidade mínima de tecnologia que seja adequada para resolvê-los (Hofstader, 2008). Assim, o arquiteto não deve se deixar levar pela sua curiosidade e vontade de experimentar novas tecnologias se elas não forem necessárias para o desenvolvimento da solução.

Por fim, é interessante que arquitetos tenham também conhecimento de programação para que possam entender possíveis problemas e também identificar soluções para eles. Além disso, arquitetos sem adequado conhecimento de programação ficam afastados do time de desenvolvimento e podem não ser capazes de ajudá-los com problemas relativos às tecnologias adotadas no projeto (Unde, 2008). E mais, pode ser necessário que o arquiteto prototipe elementos de seu *design* para garantir que esses elementos se comportarão como previsto, ou mesmo testar suas ideias antes de passá-las para o time de desenvolvimento. O trabalho do arquiteto não é apenas garantir que as suas ideias podem ser implementadas, mas também garantir que elas podem ser implementadas *efetivamente* e que o processo de criação dessa implementação é eficiente (Taylor *et al.*, 2010).

4. Habilidades de gerência de projeto:

   Arquitetos fazem nada menos que a gerência de projeto sob o ponto de vista da tecnologia. Nesse âmbito, eles costumam realizar estimativas, escolher metodologias de desenvolvimento e planejar juntamente com os gerentes de projeto. Além disso, eles precisam guiar seus times para seguirem o processo e manterem a disciplina. Ele deve entender tanto de metodologias de desenvolvimento, como RUP, CMMI,

métodos ágeis, entre outros, quanto de metodologias e/ou *frameworks* arquiteturais (Unde, 2010).

5. Experiência:

   Um arquiteto deve ter uma experiência extensiva porque ele deve se esforçar para balancear da forma mais eficaz os métodos, o alcance e a intuição para o problema em questão. Para problemas novos ele precisará fazer uso da sua intuição, mas se for um problema ou variação de um problema conhecido ele poderá usar tanto métodos específicos quanto soluções reutilizadas de acordo com a sua experiência (Taylor *et al.*, 2010). Assim, os arquitetos precisam ser expostos a projetos de escopos e tamanhos variados usando diversas tecnologias diferentes, para que, com essa experiência adquirida, possam reutilizar ideias, métodos e soluções em trabalhos futuros. É interessante ressaltar a importância do tamanho do projeto: considerações arquiteturais para um projeto pequeno para um número limitado de usuários são totalmente diferentes daquelas direcionadas a grandes aplicações acessadas por muitos usuários em todo o planeta (Unde, 2008).

2.3.2. Aspectos não técnicos

É fato que um bom arquiteto deve ter um bom conhecimento de tecnologia, mas ele também precisa ter mais que isso. Morneau e Talley (2007) afirmam que essas habilidades técnicas tradicionais se tornaram *commodities*, uma vez que passaram a estar mais facilmente disponíveis e por custos mais baratos, e isso aumentou a importância das habilidades não técnicas dos profissionais de TI. Na verdade, as habilidades sociais e de negócio (também chamadas ñ*soft skills*ò) separam um tecnologista habilidoso de um verdadeiro arquiteto (Shirey, 2008). De fato, dos cinco pilares identificados pela IASA como fundamentos necessários para um arquiteto, 2 deles estão relacionados às *soft skills*: estratégia tecnológica de negócio e dinâmica humana (Wilt, 2010)(Shirey, 2008). Muitas empresas focam nos pilares técnicos, como *design* ou ambiente de TI, mas, durante a pesquisa da IASA, foi perguntado aos arquitetos quais eram as atividades mais importantes para eles: em segundo lugar ficou a estratégia tecnológica de negócio e em primeiro lugar as dinâmicas humanas. Ou seja, os dois pilares de *soft skills* são aqueles considerados mais importantes pelos próprios arquitetos (Wilt, 2010).

O pilar de estratégia tecnológica de negócio está relacionado com as habilidades que um arquiteto deve ter para identificar, fazer o *design*, planejar e apoiar um ambiente tecnológico que proveja a base de vantagens competitivas para a companhia (Wilt, 2010), ou seja, ele deve usar a tecnologia para apoiar e favorecer o negócio da empresa. Já o pilar de dinâmicas humanas, considerado o mais importante pelos arquitetos consultados no estudo da IASA (Wilt, 2010), engloba aquelas habilidades relacionadas a gerenciar e influenciar as pessoas e suas inter-relações no contexto de um projeto ou ambiente de TI (Wilt, 2010).

Para Shirey (2008), uma solução técnica eficiente também requer 3 diferentes *soft skills*: alinhamento com o negócio, percepção de diferentes perspectivas[4] e comunicação. Esse esquema é representado na Figura 2.2 a seguir.

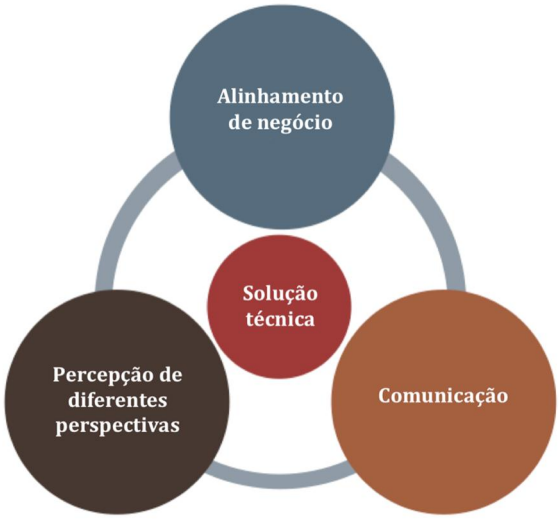

Figura 2.2 *Soft skills* necessárias para soluções técnicas de sucesso (Shirey, 2008)

Dessa forma, um arquiteto de sucesso deve ser capaz de combinar suas habilidades técnicas e não técnicas para assim aumentar sua efetividade dentro dos projetos (Shirey, 2008). A seguir serão enumeradas e brevemente descritas as atividades não técnicas mais comumente relacionadas ao papel de arquiteto de TI:

1. Liderança:

   Para Unde (2008) o título de arquiteto é uma designação de liderança. Os arquitetos tomam muitas decisões que são cruciais para o sucesso do projeto e, para implantá-las, precisam convencer várias pessoas diferentes de que suas decisões estão corretas, o

[4]do inglês *perspective awareness*

que é confirmado no estudo de Grinter (1999) para arquitetos de software e aqui estendido para arquitetos de TI. Assim, para alcançar e implantar essas ideias, os arquitetos precisam ter um bom conhecimento do ambiente político da empresa bem como a habilidade de conseguir o ñ*buy-in*ò de todos os *stakeholders* envolvidos para assim mover o projeto adiante (Unde, 2008). Entretanto, em alguns casos conseguir com que os *stakeholders* adotem suas soluções não é suficiente, o arquiteto precisa encorajá-los e fazer com que eles se empolguem com o projeto, mesmo sem possuir autoridade explícita sobre os *stakeholders* (Taylor *et al.*, 2010). Para isso o arquiteto precisa estar confiante o suficiente no sucesso do projeto e nas suas decisões, deixando transparecer a convicção nas suas ideias e sabendo enfrentar críticas negativas (Unde, 2008)(Taylor *et al.*, 2010).

Outras características necessárias a um arquiteto para que ele exerça a liderança inerente de seu papel são: mostrar-se pronto para assumir a responsabilidade por qualquer problema técnico, estar pronto para articular as razões técnicas para as decisões de *design*, ser capaz de desenvolver a arquitetura detalhada do projeto, reconhecer contribuições de outros, evitar auto engrandecimento, saber contornar obstáculos e proteger a equipe de desenvolvimento de pressões políticas (Unde, 2008)(Taylor *et al.*, 2010).

2. Alinhamento do negócio:

   Este quesito trata do pilar da IASA indicado pelos arquitetos como o segundo mais importante (Wilt, 2010). Os arquitetos precisam pensar em como a tecnologia se alinha com os objetivos de negócio, uma vez que todas as decisões, artefatos e elementos de *design* devem estar ligados a um requisito de negócio (Wilt, 2010).

   As habilidades ligadas a esta ñ*soft skill*ò não estão tão relacionadas com TI, elas estão mais relacionadas com a área de negócios da empresa, sendo muito parecidas com habilidades como as apresentadas em cursos de MBA. Um arquiteto precisa de habilidades de negócio começando com competências básicas como o entendimento sobre a forma como a organização em que ele trabalha se coloca no ambiente competitivo. Ele também precisa não apenas conhecer os cálculos financeiros, mas também saber usá-los para justificar suas decisões arquiteturais, especialmente ao negociar com CIOs e CEOs (Wilt, 2010). Para manter um diálogo significativo com os clientes e estabelecer uma relação de confiança com eles, o arquiteto precisa usar o

vocabulário de negócios e utilizar exemplos que sejam do domínio deste tipo de *stakeholder* (Unde, 2008).

Compreender bem as verdadeiras necessidades do negócio inclui também compreender os motivos de certas pressões. Shirey (2008) cita o exemplo de uma empresa onde trabalhou que exigia um tempo muito curto para a entrega do projeto e, ao investigar o negócio para entender o porquê daquele tempo, percebeu-se que se a empresa não tivesse o produto naquele período ela seria obrigada a fechar. Assim, foi necessária uma série de concessões para que o produto fosse entregue no tempo que o negócio exigia (deixando bem claro o que tais concessões provocariam no futuro) (Shirey, 2008).

Por fim, esse aspecto inclui atividades relacionadas a definição das intenções do negócio, estratégia e *roadmaps* empresariais (Kralj, 2008) e trabalhar com inovação e tendências de mercado (Wilt, 2010).

3. Comunicação:

Na pesquisa da IASA os arquitetos apontaram as dinâmicas humanas como o pilar mais importante para o seu trabalho. Tal pilar, como anteriormente descrito, está associado com a gerência de pessoas e suas inter-relações no contexto do projeto ou ambiente de TI (Wilt, 2010). Este pilar está relacionado também com o aspecto de liderança descrito acima, mas também está muito ligado à habilidade de comunicação. De fato, uma das principais responsabilidades do arquiteto é comunicar a solução definida por ele para *stakeholders* técnicos e não técnicos, precisando interagir com diversos tipos de *stakeholders*, que podem ser internos ou externos (Hofstader, 2008)(Unde, 2008)(Taylor *et al.*, 2010). Essas diferentes pessoas que ocupam diferentes papéis esperam que os arquitetos ajam de diferentes formas. Por exemplo, os executivos desejam ver a solução sob uma visão de negócio, explicando investimentos, retornos e benefícios; enquanto desenvolvedores estão mais interessados com aspectos técnicos sobre a implementação da tecnologia que usarão em seu trabalho. O arquiteto precisa compreender as necessidades dos diferentes *stakeholders* e modificar o seu estilo de articulação, sua forma de comunicar e o conteúdo que passará de acordo com o grupo com quem interagir (Unde, 2008)(Wilt, 2010). Ao mesmo tempo em que ele usa diferentes linguagens e foca em diferentes aspectos conforme interage com os diferentes *stakeholders*, ele também precisa

garantir a integridade da arquitetura, evitando que as decisões tomadas sejam burladas e que haja uma degradação da arquitetura proposta (Taylor *et al.*, 2010).

Grinter (1999) afirma também que, além de interagirem com pessoas de departamentos diferentes na empresa, os arquitetos vivem em dois contextos diferentes: clientes e organização desenvolvedora. Por esse motivo, eles devem ajudar na difusão de informações no projeto agindo como o que ela chama de ñ*Boundary Spanners*ò, isto é, pessoas que interagem com grupos diferentes transferindo informações sobre o andamento do projeto. Dessa forma, os arquitetos devem usar sua rede de contatos para difundir tais informações e auxiliar na coordenação de grupos heterogêneos com prioridades e agendas diferentes (Grinter, 1999), ajudando assim a manter uma visão coerente e única do projeto para todos esses diferentes grupos de *stakeholders* (Taylor *et al.*, 2010).

Dessa forma, os arquitetos precisam tanto receber informações de vários grupos diferentes quanto repassar tais informações para os outros grupos de uma forma que atenda as suas expectativas. Assim, é importante que eles escutem os usuários de negócio para compreender seu problema de negócio, a gerência sênior para compreender a solução mais viável, e os desenvolvedores para entender os possíveis problemas de implementação. E, ao mesmo tempo, é importante que eles articulem efetivamente a solução para os usuários de negócio para gerar o ñ*buy-in*ò, para a gerência sênior conseguir financiamento e apoio, e para os desenvolvedores para que eles compreendam como implementar a arquitetura definida (Unde, 2008).

Considerando toda a importância dessa comunicação, é fácil perceber que os arquitetos não podem se isolar dentro do projeto. Eles devem lembrar que são parte de um todo maior e que faz parte do seu trabalho perguntar, observar, escutar e comunicar. Através desses processos o arquiteto consegue, além de *feedback* das suas decisões, informações muito úteis para a própria tomada dessas decisões (Taylor *et al.*, 2010).

4. Percepção de diferentes perspectivas:

   Durante o desenvolvimento do projeto, é comum que as diferentes equipes tenham diferentes visões sobre o mesmo. Isso é problemático, pois essas visões podem ignorar alguns aspectos de negócio ou mesmo serem muito específicas da área de um grupo de

stakeholders, ignorando aspectos referentes a outros grupos. Por exemplo, os times de desenvolvimento tendem a não considerar aspectos fora do seu escopo e focar em detalhes de implementação que podem ser muito bons pontualmente mas que, quando se olha o todo, podem trazer prejuízos. Isso é mais notado no grupo de desenvolvimento especialmente, pois este tem a tendência de se isolar dos outros grupos do projeto e por isso acaba ignorando aspectos importantes a esses outros grupos (Shirey, 2008). Dessa forma, é papel do arquiteto tentar manter uma visão única da arquitetura e, para isso, é fundamental que a arquitetura que ele definir contemple as necessidades dos vários grupos, caso contrário essa visões diferenciadas poderão causar vários problemas no futuro. Esse aspecto está também relacionado com o papel de ñ*Boundary Spanners*ò que os arquitetos devem realizar, de acordo com Grinter (1999).

2.3.3. Diferentes tipos de arquitetos

Como é possível perceber, existem muitos aspectos com os quais um arquiteto deve se preocupar. Segundo Wilt (2010) e a própria definição de arquitetura de TI da IASA, um arquiteto de TI é um estrategista de tecnologia para o negócio. Essa definição é ampla e engloba vários aspectos, significando que um arquiteto de TI é ao mesmo tempo (Wilt, 2010):

a) quem modela o negócio a partir de uma perspectiva de TI;

b) quem dirige um processo rápido e iterativo de criação de código;

c) quem faz *design* e engenharia avançada;

d) quem seleciona os *frameworks* e produtos de um projeto; e

e) quem seleciona os serviços e provedores certos.

Considerando esses muitos aspectos e também o fato de que o arquiteto interage com *stakeholders* de grupos muito diferentes (como desenvolvedores, analistas, gerentes, marketing etc), Unde (2008) afirma que o papel de arquiteto é ambíguo, uma vez que cada tipo de *stakeholder* com quem o arquiteto interage espera que ele aja de uma forma diferente, o que ainda varia de organização para organização, tornando-o o um papel bastante ambíguo (Unde, 2008)(Akenine, 2008)(Hofstader, 2008). Além disso, é bastante difícil que uma única pessoa tenha habilidades suficientes em todos os aspectos descritos nas seções 2.3.1 e 2.3.2, o que inclui conhecimento de negócios, desenvolvimento, tecnologias, habilidades de gerência

de projetos, liderança, entre outros. Dessa forma, Unde (2008) cita que algumas organizações tentam diminuir essa ambiguidade criando diferentes tipos de arquiteto. Christensen e colegas (2009) afirmam que essa divisão poderia apenas refletir a tendência natural das pessoas de moldar papéis de acordo com seus interesses, porém existem organizações que definem explicitamente vários tipos de arquiteto que têm diferenças no balanceamento entre foco gerencial, técnico e de negócio. Tais tipos de arquitetos se complementam ao invés de descreverem diferentes abordagens para o mesmo papel, e as pessoas são selecionadas para eles de acordo com suas habilidades pessoais e interesses (Christensen *et al.*, 2009).

Tal esquema se alinha com os conceitos da IASA, sobre especializações da arquitetura (Wilt, 2010), e da ITAC, sobre as diferentes disciplinas (The Open Group, 2008). Em ambos os casos, existe um corpo de conhecimento fundamental sobre arquitetura que todos os arquitetos devem ter e, após adquirirem este conhecimento, existem especializações do papel de arquiteto que os profissionais podem seguir.

O problema é que não há padronização nesses diferentes tipos de arquitetos que as próprias organizações criaram, ou seja, cada uma delas tende a criar os seus próprios tipos com denominações diferentes, gerando uma grande variedade de papéis de arquitetura que possuem nomes diferentes mesmo quando executam tarefas similares. Em sua pesquisa, a IASA encontrou mais de 50 tipos diferentes de tipos de arquitetos espalhados pelas organizações e, após mapear esses tipos através dos artefatos que eles produzem, ao final foram recomendados quatro papéis[5] de arquitetos, que serão descritos a seguir (Akenine, 2008).

2.3.3.1. Arquiteto *enterprise*

O primeiro papel apresentado pela IASA é o de arquiteto *enterprise*. Este papel tem como missão apoiar a estratégia de negócio da organização com informações e soluções de TI (Akenine, 2008). Ele pode ser representado por uma pessoa ou um grupo e é responsável pela estratégia geral relacionada às capacidades de TI tanto quanto garantir que a arquitetura de TI tenha um custo efetivo. Este papel reporta geralmente para um CIO ou arquiteto chefe (Akenine, 2008).

[5] A partir daqui, ao falar especificamente da pesquisa da IASA será utilizado o termo ñpapéis de arquitetosò, uma vez que o termo em inglês usado no estudo é ñ*architect roles*ò. Entretanto, para evitar confusão no significado do texto, para qualquer outra menção a diferenciações do trabalho do arquiteto, como, por exemplo, no Capítulo 4 quando é descrito o trabalho que os arquitetos fazem nas empresas estudadas, o termo ñtipos de arquitetoò será utilizado.

Com suas atividades, o arquiteto *enterprise* irá garantir que os investimentos em TI estão alinhados à estratégia de negócio da empresa, sendo responsável por estratégias de diferentes níveis na organização, como padrões de impacto global e estratégias relacionadas a assuntos como segurança e infraestrutura. Akenine (2008) usa a analogia de um urbanista que, usando de estratégias, planejamento e regulamentações, é responsável por diferentes serviços de uma cidade que devem funcionar juntos efetivamente.

Akenine (2008) afirma que este padrão ainda é imaturo nas organizações, enquanto Unde (2008) afirma que em algumas organizações esse papel é unido com o papel de CIO e é chamado de arquiteto chefe.

2.3.3.2. Arquiteto de negócio

Este papel, assim como o primeiro, tem o foco na organização como um todo. Entretanto, este trabalha mais próximo à área de negócios da empresa, compreendendo em detalhes como a organização funciona e estando frequentemente envolvido em áreas relacionadas ao negócio em geral e também a melhoria de processos. Os arquitetos de negócio também sugerem melhorias na área de TI da organização, em conjunto com os arquitetos *enterprise*, uma vez que entendem como os sistemas de TI apoiam o negócio. Eles participam da modelagem de processos na empresa e apoiam os arquitetos de soluções na análise e nos requisitos de soluções tanto novas quanto já existentes. Dessa forma, apesar do foco na organização, eles também são ativos nos projetos em andamento por usarem sua influência para garantir que o projeto forneça benefícios para o negócio da organização (Akenine, 2008).

2.3.3.3. Arquiteto de soluções

Morneau e Talley (2007) citam a necessidade de um novo tipo de profissional de TI, chamado arquiteto de soluções. Este seria a pessoa que planeja, cria e supervisiona ou participa da construção de produtos, processos e serviços de TI. Ele participa do *design*, que trata do planejamento e criação de modelos de produtos e serviços de TI, e da construção, que envolve desenvolvimento, teste e entrega do *design*.

Esta definição de Morneau e Talley está associada com a definição que a IASA dá para o referido papel. Para a IASA, o arquiteto de soluções é quem irá balancear os requisitos funcionais e não funcionais e estabelecer as prioridades. Ele alinha novas soluções com os princípios arquiteturais considerando os padrões e a integração dentro da organização. Seu

objetivo é o sucesso do projeto atual, além de se responsabilizar tanto pelo alinhamento do projeto com os princípios arquiteturais da organização quanto pelo reuso de capacidades já existentes na organização. Akenine (2008) cita que há pessoas que definem este papel como uma evolução natural do papel tradicional de arquiteto de sistemas, uma vez que modificando o foco de sistemas para soluções são geradas novas competências e responsabilidades.

2.3.3.4. Arquiteto de software

Por fim, o arquiteto de software ï também chamado de arquiteto técnico (Unde, 2008) ï trabalha com a estrutura e *design* de sistemas de software, trabalhando tanto com requisitos funcionais quanto com atributos de qualidade, como performance e reusabilidade. Alguns dos atributos de qualidade são compartilhados com o arquiteto de soluções. Seu foco também é o projeto atual, porém a diferença é que o arquiteto de soluções possui um foco mais amplo para reuso de ativos existentes e políticas e regulamentações, enquanto o arquiteto de software possui um conhecimento mais aprofundado na tecnologia (Akenine, 2008).

2.4. A Arquitetura de software

Como se pode perceber existem diferentes tipos de arquitetos que contrastam com o que a literatura define como arquiteto, normalmente chamado de arquiteto de software (Christensen *et al.*, 2009). Além disso, analisando os papéis recomendados pela IASA é possível notar que 2 destes papéis (os arquitetos *enterprise* e de negócios) possuem um foco organizacional enquanto os 2 outros (os arquitetos de soluções e de software) focam nos projetos em que estão alocados no momento. Os dois papéis mais organizacionais estão mais relacionados com estratégias de gerenciamento, conhecimento do negócio, padrões organizacionais, questões de mercado; enquanto os outros dois papéis possuem um foco mais relacionado com o que se chama de arquitetura de software.

O que se observa é que alguns arquitetos passam a maior parte do tempo envolvidos diretamente com a equipe de desenvolvimento, enquanto outros ocupam-se mais da arquitetura geral (Christensen, 2009). Taylor *et al*. (2010) definem arquitetura de software como o conjunto das principais decisões de *design* tomadas durante o desenvolvimento do sistema e qualquer evolução do mesmo. Nessa definição estão incluídas atividades relacionadas tanto com a criação da visão geral da arquitetura quanto com o refinamento da mesma e o trabalho próximo aos desenvolvedores. Portanto, pode-se encontrar nesta definição as atividades realizadas pelos arquitetos de soluções e de software recomendados pela IASA

(Akenine, 2008). Dessa forma, se torna interessante que a arquitetura de software seja discutida.

A arquitetura de software, segundo Taylor *et al*. (2010), como descrito antes, trata de todo o conjunto das principais decisões que irão reger o *design* do software durante o seu desenvolvimento. Por sua vez, Ovaska *et al.* (2003) afirmam que a visão estrutural de Conway é a mais amplamente aceita para definição de arquitetura de software: ñA visão estrutural define arquitetura como a estrutura dos componentes de um sistema, seus inter-relacionamentos, e princípios e *guidelines* que influenciaram seu *design* e evolução através do tempoò. Já Bass *et al*. (2003) definem arquitetura de software como ña estrutura ou as estruturas do sistema, que compreendem elementos de software, as propriedades desses elementos visíveis externamente, e os relacionamentos entre elesò.

A partir das diferentes definições anteriores, pode-se concluir que um dos aspectos importantes da arquitetura de software é a estrutura do sistema que será desenvolvido. Tal estrutura engloba tanto a visão geral (foco dos arquitetos de soluções) quanto o refinamento da mesma (foco dos arquitetos de software). Esse é um aspecto bem conhecido e discutido na literatura, por exemplo (Fowler, 2002) e (Bass *et al.*, 2003), e abrange modelos arquiteturais, padrões de projetos, métodos de avaliação de arquitetura etc. Este é um aspecto importante, mas existe mais na arquitetura de que puramente sua estrutura: é preciso considerar também aspectos sociais e organizacionais relativos à arquitetura de software, pois estes influenciam muito na forma como ela é desenvolvida, inclusive tecnicamente.

A arquitetura de software é um ponto chave de qualquer desenvolvimento e sua interpretação é útil para o trabalho de diferentes grupos de *stakeholders*. Entretanto, além de não existir uma definição padrão para a arquitetura de software, os diferentes *stakeholders* não a enxergam da mesma forma (Smolander, 2002). Smolander (2002) usou o termo metáforas de arquitetura para diferenciar as conotações que os *stakeholders* dão para ela, sendo que os significados variam de acordo com a posição na hierarquia organizacional e o *background* do próprio *stakeholder*. Para Smolander, a arquitetura de software é uma metáfora onde nós compreendemos a estrutura de um sistema de software em termos de construções que nós conseguimos captar mais facilmente. Assim, ele definiu quatro metáforas de acordo com os resultados do seu estudo: arquitetura como *blueprint*, como literatura, como linguagem e como decisão (Smolander, 2002). Cada uma destas metáforas é discutida a seguir.

A primeira metáfora, ñarquitetura como *blueprint***ò**, refere-se à estrutura do sistema a ser implementado. Ela é mais utilizada por implementadores e programadores e trata da implementação de alto nível do sistema. Com isso, seu objetivo é transferir informações explícitas dos arquitetos para os *designers* e outros engenheiros de software utilizando de linguagens de descrição de arquitetura.

Já para ñarquitetura como literaturaò a arquitetura reside na documentação e referencia a arquitetura para futuros leitores. Dessa forma, ela está ligada à documentação das estruturas técnicas e à transferência de informações através do tempo, formando um conjunto de soluções feitas no passado que visam reuso de componentes e *design*. Esta metáfora não é específica para um grupo de *stakeholders.*

A terceira metáfora, ñarquitetura como linguagemò, vê a arquitetura como a linguagem para descrever um conceito comum sobre o sistema. Ela visa uma comunicação efetiva entre os diferentes grupos de *stakeholders,* logo não possui muitos detalhes e formalidades. É uma metáfora mais voltada para clientes ou aspectos de negócios, envolvendo muitas vezes gerentes ou pessoas com responsabilidades de venda. Um ponto interessante encontrado por Smolander é o fato de os desenvolvedores não enxergarem essa metáfora, pois para eles o aspecto técnico é mais importante.

Por fim, para a última metáfora, ñarquitetura como decisãoò, a arquitetura é a decisão e a base para decisões sobre o sistema a ser implementado. Ela trata de recursos, força de trabalho e suas habilidades, custo, acordos, estratégias etc. A descrição da arquitetura se torna a tomada de decisão racional sobre recursos (pessoas, custo) e estratégias e visa planejar, avaliar alternativas e selecionar e tomar decisões sobre as soluções técnicas. Geralmente, a arquitetura é apresentada sem muitos detalhes e formalidades, sendo voltada para gerentes e planejadores de recursos também como uma ferramenta para lidar com riscos.

Assim, enxergar a arquitetura de software como apenas a estrutura técnica do sistema em desenvolvimento é enxergar apenas uma parte da importância da mesma. E, apesar disso, Smolander afirma que a maioria dos estudos foca justamente na metáfora de *blueprints*, e poucos tratam da metáfora de ñarquitetura como linguagemò (Smolander, 2002). Entretanto, esta última pode ser muito útil para construir o entendimento comum sobre as interdependências entre os trabalhos das equipes. Caso tal entendimento comum não exista, podem acontecer problemas relacionados a suposições que as equipes precisam fazer sobre o trabalho umas das outras, como identificado por Grinter (1999). Indo mais além, Ovaska *et*

al.(2003) sugerem que a arquitetura seja utilizada como meio de coordenação, especialmente em projetos desenvolvidos de forma distribuída. Com isso, pode-se perceber que um importante aspecto relacionado à arquitetura de software trata da coordenação que ela pode favorecer. Tal aspecto será discutido a seguir.

2.4.1. Coordenação e arquitetura de software

Problemas de coordenação são uma das principais causas de atraso e estouro de orçamento (Curtis *et al.*, 1988)(Ovaska *et al.*, 2003). Para Ovaska *et al.* (2003) uma das formas de ajudar a evitar tais problemas é utilizar a arquitetura como um mecanismo para auxiliar na coordenação, mas isso vai depender de pessoas chave.

Em 1968, Conway relacionou a arquitetura com a estrutura de comunicação da organização (Conway, 1968) e em 1972, Parnas tratou do encapsulamento e ocultamento de informação, afirmando que limitar as interações entre os módulos (que ele define como atribuições de responsabilidades) pode permitir seu desenvolvimento paralelo por times separados com o mínimo de interação entre eles (Parnas, 1972). Esses dois trabalhos já indicavam uma relação entre a arquitetura e a coordenação das atividades dentro de um projeto.

Por coordenação, entende-se o gerenciamento das interdependências entre as atividades (Ovaska *et al.*, 2003). Com isso, se não há interdependências não há o que coordenar (Ovaska *et al.*, 2003). Dessa forma, o papel da arquitetura na coordenação já começa na criação de seus módulos. Como Parnas (1972) indicou, é necessário que o *design* seja modular, especialmente no caso de times distribuídos, e deve-se atender à Lei de Conway (Conway, 1968) usando esse *design* modular para dividir o trabalho entre os diferentes grupos e/ou *sites* (Herbsleb e Grinter, 1999).

Este é o ponto importante para Bass *et al.* (2007), pois para eles não são as estruturas organizacionais nem as decisões arquiteturais isoladas que causam problemas, mas sim a falta de alinhamento entre essas áreas. Em seu trabalho, eles chamam de ñdesalinhamento arquiteturalò[6] o fato de atribuir tarefas a times que não são capazes de efetivamente gerenciar a coordenação (Bass *et al.*, 2007). Como isso gera problemas, alguns arquitetos explicitamente consideram alocação de trabalho quando decompõem o sistema. E também empregam um esforço extra para desacoplar (ñ*looselycouple*ò) ou minimizar a complexidade

[6]do inglês *architecturalmisalignment*

e o número de invocações entre componentes destinados a times que tem habilidade limitada de coordenação (Bass *et al.*, 2007).

Ovaska *et al.* (2003) observaram que problemas de coordenação no *design* da arquitetura resultam em uma descrição arquitetural pobre ou inexistente e em diferentes interpretações sobre a arquitetura pelos membros do projeto. O segundo ponto é especialmente interessante uma vez que é necessário que todos os *stakeholders* envolvidos possuam uma mesma visão sobre o sistema que estão desenvolvendo. Deve haver uma pessoa, que Ovaska *et al.* (2003) chamam de arquiteto chefe, para comunicar as estruturas e soluções do sistema para assim conseguir esse entendimento comum da arquitetura e ajudar a coordenar o trabalho de desenvolvimento (Ovaska *et al.*, 2003). Tal trabalho deve ser realizado de forma que os diferentes grupos sejam considerados e informados, especialmente em desenvolvimento distribuído de software.

Os arquitetos interagem naturalmente com diversos tipos de *stakeholders*. As interações mais comuns do arquiteto com outros papéis são com os gerentes e engenheiros de software, o que inclui testadores, engenheiros de requisitos e programadores (principalmente). Com os engenheiros de software os arquitetos devem ter uma relação cooperativa. Por exemplo, os programadores também trabalharão como validadores da arquitetura, uma vez que se uma decisão de *design* tiver uma consequência não prevista pelo arquiteto provavelmente os programadores serão os primeiros a notar (Taylor *et al.*, 2010). Por sua vez, arquitetos e gerentes devem trabalhar juntos para alinhar seus objetivos. Gerentes devem ajustar suas expectativas do projeto de acordo com considerações técnicas da arquitetura, enquanto os arquitetos devem ajustar as suas próprias expectativas e a própria arquitetura de acordo com a realidade da organização (Taylor *et al.*, 2010).

Ainda relacionado ao trabalho de arquitetura de software, os arquitetos ainda interagem, embora em menor escala, com outros *stakeholders*. Por exemplo, há interação também com o que em algumas organizações é chamado de departamento de *marketing* (Taylor *et al.*, 2010). Esses departamentos pesquisam a situação do mercado para determinado produto identificando o que os consumidores precisam ou podem vir a precisar. As informações provenientes deles ajudarão os arquitetos a criar *designs* que aproveitem as condições do mercado e satisfaçam os clientes. Similar a essa relação existe também a interação arquiteto-cliente/usuário, uma vez que o objetivo principal de um sistema é satisfazer as necessidades dos clientes/usuários (Taylor *et al.*, 2010).

As barreiras de comunicação entre os times ou grupos podem ser diminuídas com um arquiteto que aja como *Boundary Spanner* entre eles, uma vez que um *Boundary Spanner* traduz necessidades do cliente em termos que podem ser entendidos por desenvolvedores de software (Ovaska *et al.*, 2003). Assim, como os arquitetos interagem naturalmente com vários tipos de *stakeholders*, eles podem e devem utilizar essas interações para ajudar na coordenação dentro do projeto, agindo assim como *BoundarySpanners* (Grinter, 1999)(Ovaska *et al.*, 2003). Além disso, Unphon e Dittrich (2010) identificaram em seu estudo que o arquiteto é a principal fonte de informações dentro de um projeto. É mais comum que as informações sejam ñarmazenadasô com eles do que em documentações. Assim, eles definiram o termo de ñarquitetura ambulanteò[7] que significa que o arquiteto de software é a pessoa que comunica a arquitetura de software para os desenvolvedores e os problemas encontrados para os *stakeholders* certos. Com isso, percebe-se que não somente a arquitetura de software influencia na coordenação, mas também os arquitetos exercem um papel muito importante nesse sentido. Eles devem usar sua rede de comunicação primeiramente para coletar as informações necessárias para construir um sistema que atenda as necessidades existentes, mas também que considere a estrutura da própria organização. Com isso deve-se criar um *design* de arquitetura modular de forma que as tarefas possam ser divididas entre os times favorecendo as interações e coordenações entre eles. Além disso, o papel de arquiteto deve estar presente em todo o período do projeto para ajudar a manter o entendimento comum sobre a arquitetura, interagindo e difundindo informações para os *stakeholders* corretos, para assim não ocorrerem problemas relacionados a divergências de visões.

[7] do inglês *walking architecture*

CAPÍTULO 3

3. Metodologia

3.1. Visão geral

Como descrito no Capítulo 1, este trabalho tem como objetivo entender como os diferentes tipos de arquitetos de TI realizam suas atividades nas empresas. Para atingir este objetivo optou-se pela realização de um estudo qualitativo, uma vez que este tipo de estudo baseia seus resultados nas observações das pessoas que realizam a atividade que é objeto do estudo (Strauss e Corbin, 2008). Assim, esta pesquisa foi realizada principalmente com arquitetos de TI profissionais que trabalham em empresas de TI, uma vez que eles formam o grupo de interesse da pesquisa. No início também se estendeu a pesquisa a outros papéis que influenciam e/ou são influenciados pela arquitetura durante o desenvolvimento, como por exemplo líderes de projeto e desenvolvedores, para assim obter uma visão ampla do fenômeno estudado.

Um método qualitativo de investigação foi adotado, por serem os indivíduos que realizam as atividades, os arquitetos de TI, as pessoas que efetivamente dão significado ao domínio, ou seja, são eles que possuem o conhecimento necessário para interpretar as ações de forma relevante para o contexto (Seaman, 2008). Um método qualitativo também se faz necessário porque o estudo proposto é exploratório, isto é, *a priori,* não existem hipóteses definidas a serem testadas, ou ainda, não se sabe como a atividade em questão é feita na prática.

Optou-se por entrevistas semiestruturadas (Singer *et al.*, 2008) como método de coleta de dados, pois através delas é possível identificar a visão dos entrevistados sobre o assunto estudado, ou seja sobre como exercem suas atividades na prática. Tais entrevistas foram transcritas e analisadas usando técnicas da teoria fundamentada em dados (Strauss e Corbin, 2008), de forma a, ao final, montar uma teoria sobre o fenômeno estudado. Segundo esta teoria, durante a execução de métodos qualitativos, etapas de coleta e análise de dados devem ser alternadas para que os resultados da análise possam direcionar a nova etapa de coleta de dados, sugerindo focos e novas abordagens que poderão modificar ou corroborar os resultados obtidos em um dado momento. Dessa forma, foram realizadas 4 etapas de coleta de dados

(sempre intercaladas com etapas de análise) com profissionais de empresas desenvolvedoras de software que desempenhavam funções relacionadas a arquitetura de TI (incluindo a arquitetura de software) da empresa.

3.2. Pesquisa Qualitativa

Uma pesquisa qualitativa é aquela que tenta entender o significado ou a natureza da experiência de um determinado grupo de pessoas. É qualquer tipo de pesquisa que produza resultados que não podem ser alcançados por meio de procedimentos estatísticos ou similares (Strauss e Corbin, 2008). Como descrito anteriormente, foram estudados neste trabalho os arquitetos de TI e o trabalho que eles realizam, ou seja, o foco foi um grupo de pessoas e as tarefas que elas realizam a partir do ponto de vista delas, pois, nesse caso, a ação só adquire significado entre as pessoas que compartilham o domínio.

Segundo Seaman (2008) a principal vantagem de realizar pesquisa qualitativa está no fato de que métodos qualitativos forçam o pesquisador a se aprofundar na complexidade do problema em vez de abstraí-la. Com isso, os resultados são mais ricos e mais informativos, ajudando a responder questões que envolvem variáveis que são difíceis de quantificar, como, por exemplo, características humanas como motivação, percepção e experiência (Seaman, 2008).

Segundo Strauss e Corbin (2008) há três componentes principais nesse tipo de pesquisa: dados, procedimentos e relatórios. Os dados podem vir de várias fontes diferentes, como entrevistas, observações, documentos, filmes ou gravações em vídeo. Os procedimentos são usados para interpretar e organizar os dados. Eles, geralmente, têm o intuito de atribuir conceitos e reduzir os dados, elaborando categorias que são relacionadas através de declarações proposicionais. Esse processo de atribuir, reduzir, elaborar e relacionar conceitos é chamado de codificação. Por fim, relatórios escritos ou verbais são utilizados para apresentar os resultados obtidos (Strauss e Corbin, 2008).

A seguir, serão melhor descritos os métodos de coleta e análise de dados que foram utilizados nessa pesquisa.

3.2.1. Método de coleta de dados

Um dos métodos de coleta de dados mais comuns em métodos de pesquisa qualitativos é a condução de entrevistas, abordagem que foi usada neste trabalho. As técnicas de entrevistas e

questionários são classificadas por Singer *et al.* (2008) como diretas (aquelas nas quais o pesquisador tem um envolvimento direto com a população participante) e são utilizadas por pesquisadores quando o objetivo é entender informações gerais (incluindo opiniões) sobre processos, produtos, conhecimento pessoal, etc. Essa técnica de coleta de dados gera um volume de dados que vai de pequeno a grande, e também pode ser usada para coletar requisitos e fazer avaliações (por exemplo de aceitabilidade de produtos) (Singer *et al.*, 2008).

Entrevistas vão de estruturadas a não estruturadas, variando o nível de controle que o pesquisador tem (Seaman, 2008). No caso das estruturadas, o pesquisador tem objetivos muito específicos fazendo com que suas perguntas também sejam muito específicas, para que foquem em conseguir o tipo de informações que ele procura. Isto é o oposto das entrevistas não estruturadas, onde o objetivo é coletar o máximo de informações possível para um tópico definido de forma ampla. Geralmente, entrevistas estruturadas não permitem a descoberta de tipos de informação que não haviam sido previstas, porém entrevistas completamente não estruturadas são muito custosas (Seaman, 2008)(Singer *et al.*, 2008). É por isso que muitos pesquisadores adotam entrevistas semiestruturadas, que compreendem um misto de perguntas específicas, ou fechadas, com perguntas abertas, permitindo assim elicitar os tipos de informações de interesse do pesquisador e também aqueles inesperados (Seaman, 2008).

Uma entrevista semiestruturada possui um guia que consiste de uma lista de perguntas e lembretes apenas para garantir que todos os tópicos de interesse serão abordados. As perguntas não são tão fixas e formais como em uma entrevista estruturada, mas possuem um controle maior que o existente em entrevistas não estruturadas (Dewalt e Dewalt, 2009). A entrevista segue um fluxo mais parecido com uma conversa e novas perguntas podem ser desenvolvidas conforme o pesquisador aprende novas informações (Singer *et al.*, 2008)(Seaman, 2008). O guia pode conter também anotações sobre como o pesquisador deve conduzir as entrevistas de acordo com as circunstâncias (Seaman, 2008). Os guias de entrevistas utilizados neste trabalho estão disponíveis nos Apêndices B, C e D.

3.2.2. Método de análise de dados

Em uma pesquisa qualitativa a análise dos dados é dita interpretativa, devendo-se utilizar, portanto, métodos de análise qualitativos. A análise qualitativa é o processo não matemático de interpretação, que tem por objetivo descobrir conceitos e relações nos dados brutos e de organizar tais conceitos e relações em um esquema explanatório teórico (Strauss e Corbin,

2008), isto é, um esquema que explane/explique os dados coletados, ou seja, uma teoria que seja fundamentada pelos dados obtidos durante as outras etapas da pesquisa, uma vez que, neste tipo de análise não se tem inicialmente uma hipótese sobre os dados a ser testada (Strauss e Corbin, 2008).

Geralmente, pesquisas qualitativas geram uma grande quantidade de dados que devem ser analisados. Para isso, o conjunto de dados deve ser reduzido para um formato compreensível, o que é tradicionalmente feito através do processo de codificação (Singer *et al.*, 2008). Tal processo consiste de usar os objetivos da pesquisa como um guia e assim desenvolver um esquema para categorizar os dados. Depois dessa categorização, os dados podem ser analisados para prover uma caracterização baseada nos esquemas de codificação dos pesquisadores (Singer *et al.*, 2008).

É importante reafirmar que em uma pesquisa qualitativa as etapas de coleta e análise de dados são alternadas e que a análise começa assim que se possua um conjunto significativo de dados coletados (Seaman, 2008). Além disso, o resultado de análises preliminares pode também modificar as etapas de coleta de dados subsequentes (Seaman, 2008). Geralmente as entrevistas são gravadas e transcritas, e podem ser analisadas com o auxílio de ferramentas de análise de dados qualitativos.

A seguir aspectos da Teoria Fundamentada em dados serão brevemente descritos, uma vez que estes métodos foram utilizados nesta pesquisa.

3.2.2.1. A Teoria fundamentada em dados

A teoria fundamentada em dados consiste em criar uma teoria que seja derivada dos dados, sistematicamente reunidos e analisados por meio de um processo de pesquisa. Nessa técnica não se tem no início uma teoria preconcebida (a não ser que o objetivo seja estender tal teoria), o pesquisador começa com uma área de estudo e permite que a teoria surja a partir dos dados, conforme ele realiza as etapas de coleta e análise (Strauss e Corbin, 2008).

Nesta técnica, as etapas de coleta e análise dos dados são alternadas e quaisquer hipóteses e proposições derivadas dos dados devem ser continuamente ñverificadasò em comparação a novos dados e modificadas, estendidas ou desconsideradas de acordo com a necessidade (Strauss e Corbin, 2008).

A base da Teoria fundamentada em dados está nas codificações, que consistem de atribuir um conceito ou categoria a uma parte dos dados. Um conceito abstrai um evento, objeto, ação ou interação que é de interesse do pesquisador, enquanto categorias são agrupamentos de conceitos unidos em um grau de abstração mais alto, minimizando o número de elementos que o pesquisador precisa considerar. Assim, uma teoria é um conjunto de categorias bem desenvolvidas que são sistematicamente relacionadas entre si através de declarações de relação para formar uma estrutura teórica que explique o fenômeno estudado. Para atingir este objetivo, a teoria fundamentada em dados possui três etapas principais: codificação aberta, codificação axial e codificação seletiva (Strauss e Corbin, 2008).

Na primeira etapa, codificação aberta, os dados são micro analisados (linha por linha) para que as categorias possam ser identificadas. As categorias são criadas para minimizar o número de elementos que precisam ser considerados pelo pesquisador. A Figura 3.1 a seguir foi retirada do trabalho de Banks *et al.* (2000) que analisa do uso de cartões postais como exemplo de codificação aberta. Nela observam-se as categorias numeradas e exemplos de trechos retirados dos cartões postais relacionados a estas categorias, seguidos do número de vezes que cada categoria foi utilizada.

2. Description of especially enjoyable experience; great fun	"He continually pushes himself to the limit (and us) to have as much fun as he possibly can, a true '**fun hog**.' " [#50]	63
3. Reference to extraordinary work effort or work demands	"Was a working dynamo — couldn't keep enough jobs ahead of him!" [#120]	3
4. Relates connections to celebrity or drops name of celebrity	"Alas, Patty did not get mentioned in humorist Dave Barry's newspaper column this year …" [#98]	2
5. Mention of experience in cultural arts — fine arts museums, opera, ballet, symphony, etc.	"She is also showing quite an interest in ballet." [#51]	19

Figura 3.1 Exemplo de codificação aberta (Banks *et al.*, 2000)

Na etapa de codificação axial, as categorias criadas previamente são quebradas em subcategorias. Enquanto as categorias representam fenômenos, as subcategorias são criadas para responder perguntas sobre estes fenômenos, tais como: por que, quando, onde, como, quem e com quais consequências (Strauss e Corbin, 2008). Assim, nesta etapa, são identificadas propriedades e dimensões de cada categoria. A Figura 3.2 a seguir também foi

retirada do trabalho de Banks *et al.* (2000) para servir como exemplo de codificação axial. Nela observam-se as subcategorias da categoria ñ*Positive Experience/Adventure*ò resultantes da codificação axial e o número de trechos (*tokens*) anexados a cada uma.

1 ***POSITIVE EXPERIENCE/ADVENTURE*** (278 tokens):
2) fun experience/great time/I-we enjoyed (63)
4) celebrity experience/name drop (2)
5) cultural activities (19)
8) exciting adventure (34)
9) you'd be amazed at what I experienced (3)
61) writer/we traveled (91)
62) family member traveled (37)
78) vacation (31)

Figura 3.2 Exemplo de categorias resultantes da codificação axial (Banks *et al.*, 2000)

Finalmente, durante a codificação seletiva, as categorias mais importantes são selecionadas para compor o conjunto que será usado para descrever a teoria. A categoria mais importante é eleita (ou criada) para que a teoria seja descrita através dela (Strauss e Corbin, 2008).

Outro conceito importante da teoria fundamentada em dados é a amostragem teórica, que consiste em escolher pontos de coleta de dados para expandir, complementar ou rejeitar a teoria que está sendo construída. A amostragem é feita com base em conceitos emergentes para explorar o escopo dimensional ou as condições variadas ao longo das quais as propriedades dos conceitos variam (Strauss e Corbin, 2008).

A seguir os detalhes sobre as empresas pesquisadas e o processo de coleta e análise dos dados serão descritos e, no próximo capítulo, os resultados encontrados serão expostos.

3.3. Contexto da pesquisa

Este trabalho descreve um estudo que visou compreender como os arquitetos de TI desenvolvem suas atividades de arquitetura na indústria. Assim, 9 diferentes organizações de TI foram visitadas e a pesquisa foi conduzida nas mesmas através de entrevistas com arquitetos de TI profissionais e outros papéis relacionados. Estas empresas foram escolhidas por questões de localização, facilidade de acesso a informantes e disponibilidade para participar na pesquisa. A seguir estas 9 empresas são brevemente descritas. Todos os dados são anônimos devido a questões de confidencialidade. As informações descritas abaixo foram retiradas dos sites das empresas e de um pequeno questionário de caracterização enviado aos

informantes (Apêndice A). Infelizmente nem todos os informantes responderam ao questionário, ou então não sabiam responder a todas as perguntas, por isso não foi possível obter o mesmo conjunto de informações sobre todas as empresas.

Empresa A ï Esta empresa é uma multinacional cuja sede está localizada nos Estados Unidos. Ela possui várias filiais espalhadas pelo mundo, inclusive no Brasil, de forma que trabalha constantemente com desenvolvimento distribuído de software. É uma empresa de TI que tem um portfólio de produtos bem amplo, atendendo desde clientes pessoais até grandes organizações, oferecendo soluções completas como serviços, hardware e software. Ela possui milhões de clientes no mundo e por volta de 170.000 empregados.

Esta pesquisa foi realizada na filial (que constitui um centro de pesquisa e desenvolvimento) desta empresa localizada em Porto Alegre, Brasil. A divisão a qual os informantes pertencem possui avaliação SW-CMM nível 2 e, de acordo com um deles, seus processos já são aderentes ao CMMi nível 3, embora a empresa ainda não tenha solicitado a avaliação. Ainda de acordo com esse informante, a empresa desenvolve centenas de projetos por ano que normalmente duram de 6 a 12 meses e alocam por volta de 70 empregados. Tais sistemas são normalmente de grande porte; um dos informantes afirmou que hoje ele trabalha em um projeto que possui por volta de 1 milhão de linhas de código. Nesta empresa foram entrevistados 3 informantes, sendo que dois apenas na última etapa.

Empresa B ï A Empresa B é também uma multinacional de TI cuja sede está localizada nos Estados Unidos. Ela desenvolve, vende e fornece apoio para computadores e produtos e serviços relacionados. Entre tais produtos estão computadores pessoais, servidores, switches de rede, software e periférico de computadores. Ela possui muitas filiais pelo mundo (trabalhando frequentemente com DDS), milhões de clientes e emprega mais de 96.000 funcionários. A pesquisa começou a ser conduzida (na primeira etapa do estudo, com duas entrevistas com 3 informantes) em um centro de desenvolvimento de software desta empresa localizado na cidade de Porto Alegre, Brasil, porém não foi possível retornar a esta empresa nas outras etapas devido a compromissos dos informantes. Dessa forma, não foi possível conseguir a quantidade de informações desejada para a pesquisa.

Empresa C ï Esta é uma companhia brasileira de desenvolvimento de software especializada em *e-commerce* para o setor de varejo. Sua matriz está localizada em um Campus

Tecnológico de uma Universidade em Porto Alegre e ela também possui escritórios em Portugal. A Empresa C provê serviços de TI para diferentes áreas de expertise, para médias e grandes organizações. Ela já desenvolveu de forma distribuída dois projetos piloto, como forma de teste, porém normalmente seu desenvolvimento é colocalizado. De acordo com os 3 informantes desta empresa, hoje ela possui entre 150 e 200 empregados, 10 clientes (metade no Brasil e metade em Portugal) e, normalmente, desenvolve por volta de 50 projetos por ano, cada um tendo em média (dependendo do tipo do projeto) 100 casos de uso e entre 1000 e 2000 horas. Dependendo do tamanho do projeto, normalmente são alocados entre 5 e 15 empregados para seu desenvolvimento. A Empresa C também possui avaliação CMMi nível 3.

Empresa D ï Esta é também uma empresa brasileira cuja matriz está localizada em Porto Alegre, possuindo também outro escritório em São Paulo. As entrevistas foram conduzidas na sede em Porto Alegre com dois informantes diferentes. A Empresa D existe desde 2002 e provê serviços em desenvolvimento de software e infraestrutura. Seus serviços são oferecidos através de consultoria, fábrica de projetos, *outsourcing* e *offshoring*. Ela possui operações no Brasil e atuações no exterior, possuindo em torno de 32 clientes, sendo 30 no Brasil e 2 no exterior, incluindo clientes dos segmentos de energia, agronegócio, tecnologia da informação, transportes, logística, petroquímica e varejo. A maioria dos projetos desta empresa é feita de forma colocalizada, mas eles também trabalham eventualmente com DDS. Em média esta empresa desenvolve 130 projetos por ano com duração média entre 3 e 4 meses cada um, sendo que são alocadas geralmente entre 5 e 15 pessoas em cada projeto (do total aproximado de 140 pessoas na empresa). A Empresa D possui avaliação CMMi nível 2.

Empresa E ï Esta empresa é uma multinacional de internet, com milhões de clientes e usuários, cuja matriz está localizada na Espanha. Ela opera como um portal web e/ou provedor de acesso à internet nos Estados Unidos, Espanha e em muitos países da América Latina, entre eles Brasil, México, Chile, Peru e Argentina. Essa pesquisa foi conduzida com dois informantes da sede do Brasil, em Porto Alegre, a qual é o escritório central da América Latina. Como esta empresa precisa integrar os sistemas de todos os países envolvidos da América Latina, ela frequentemente tem de lidar com alguma forma de distribuição em seus projetos.

Empresa F ï A Empresa F é uma companhia que desenvolve soluções corporativas para cartões de crédito e opera no setor de benefícios, com produtos para RH e gestão de frotas (abastecimento e manutenção). Ela trabalha especialmente usando a internet como ferramenta para auto gerência, permitindo que seus clientes controlem limites, requisitem e cancelem cartões e emitam extratos através da web. Ela possui por volta de 500 empregados em 9 estados do Brasil.

A Empresa F tem sua própria área de TI e as entrevistas foram realizadas com dois arquitetos desta divisão, que fica localizada em Porto Alegre. Tal divisão é uma fábrica de software que provê serviços para a empresa através de um produto específico. Portanto, a área de TI tem apenas um cliente: a própria Empresa F. Esta área de TI trabalha de forma colocalizada e desenvolve um projeto principal, no qual outros projetos menores são criados de acordo com a necessidade do cliente (a própria empresa). Dessa forma este é o ñúnicoò projeto desta empresa e é desenvolvido há 5 anos.

Empresa G ï A Empresa G é uma empresa estatal brasileira provedora de soluções de TI com foco no setor bancário. Ela é controlada por um banco brasileiro e sua matriz está localizada no Rio de Janeiro. Seus produtos e serviços incluem soluções de software integradas e *outsourcing*, gerência de redes e suporte técnico. A Empresa G possui várias filiais no Brasil e esta pesquisa foi conduzida na filial localizada em Belém. Tal filial foi criada devido à demanda de um grande projeto de um grande cliente na região; dessa forma esta filial só possui um cliente e um único grande projeto de um produto que ainda está em desenvolvimento há 5 anos. Esta filial de Belém possui em torno de 230 funcionários (e todos trabalham neste projeto da empresa ocupando vários cargos como analistas, desenvolvedores, gerentes, testadores e outros), sendo que 4 são arquitetos (3 deles foram entrevistados), e seu projeto atualmente tem em torno de 17000 classes. Seu desenvolvimento também acontece todo de forma colocalizada.

Empresa H ï A Empresa H é uma empresa global de consultoria de TI com matriz localizada nos Estados Unidos e várias filiais no mundo, trabalhando frequentemente com DDS. Ela produz aplicações customizadas e provê consultoria, possuindo em torno de 200 clientes fora do Brasil. Ela possui em torno de 1600 empregados, desenvolve uma média de 40 projetos por ano e cada um com duração de aproximadamente 4 meses alocando em torno de 20 pessoas. A pesquisa foi conduzida com uma entrevista com dois informantes da filial

brasileira localizada em Porto Alegre (com apenas 11 meses de atuação), mas não foi possível extrair a quantidade de informação desejada para a pesquisa.

Empresa I ï Esta é uma empresa brasileira com matriz em Porto Alegre que possui aproximadamente 15 clientes no país. Seu foco é banco de dados e tecnologias web de software, desenvolvimento e implementação de aplicações corporativas críticas. Ela possui experiências de desenvolvimento no Brasil, e também nos Estados Unidos e Portugal através de parcerias com outras empresas. Dessa forma, essa empresa utiliza DDS apenas eventualmente. Apenas uma entrevista com um informante foi realizada nesta empresa. A Empresa I possui avaliação CMMi nível 2.

A Tabela 3.1 a seguir traz um resumo das empresas pesquisadas. É importante ressaltar que as informações (com valores aproximados) nela contidas foram retiradas dos sites das empresas e de um pequeno questionário enviado aos informantes. Tal questionário continha perguntas relativas a cada uma das colunas da Tabela 3.1 e pode ser observado no Apêndice A deste trabalho.

Tabela 3.1 Resumo das empresas pesquisadas

Empresa	N° clientes	N° empregados	DDS	Projetos/ano	Tamanho dos projetos	Pessoas / projeto	Duração dos projetos	Modelo de maturidade
Empresa A	milhões	170.000	S	centenas	1 milhão linhas de código	70	6-12 meses	CMMi 2
Empresa B	milhões	96.000	S	-	-	-	-	-
Empresa C	10	150-200	I	50	100 casos de uso	5-15	1000-2000h	CMMi 3
Empresa D	32	140	I	130	-	5-15	3-4 meses	CMMi 2
Empresa E	milhões	-	S	-	-	-	-	N
Empresa F	1	500*	N	1**	-	-	5 anos**	N
Empresa G	1	230	N	1**	17000 classes	230	5 anos**	N
Empresa H	200	1600	S	40	-	20	4 meses	N
Empresa I	15	-	I	-	-	-	-	CMMi 2

*número de empregados da empresa toda, não apenas da divisão onde a pesquisa foi conduzida.
**a empresa possui apenas um projeto que ou ainda não foi entregue (Empresa G) ou é um projeto interno (Empresa F).

A coluna DDS indica se a empresa trabalha com esse tipo de desenvolvimento, onde ñSò significa sim e ñNò significa não. O valor ñIò vem de inicial, que significa que a empresa já teve alguma iniciativa de DDS, mas ou esta se tratava de projetos piloto (Empresa C) ou, ainda assim, não é costume da empresa usar este tipo de desenvolvimento (Empresas D e I).

A maioria dos informantes (que responderam ao questionário) não soube precisar o tamanho médio dos projetos desenvolvidos; quando respondiam a este quesito normalmente

usavam termos imprecisos como ñgrandeò, ñde médio a grande porteò e o informante da Empresa D respondeu que eles ainda estão coletando métricas de tamanho. Portanto, como tais informações não constam nos sites das empresas, não foi possível obter esses dados de todas as empresas.

A seguir o processo de coleta e análise de dados nestas nove empresas será descrito.

3.4. Processo de coleta e análise dos dados

Como descrito anteriormente, entrevistas semiestruturadas e a teoria fundamentada em dados foram utilizados como métodos de coleta e análise de dados respectivamente. Tais métodos foram descritos nas seções anteriores deste capítulo (ver seções 3.2.1 e 3.2.2). Além disso, a questão de pesquisa utilizada para guiar este trabalho foi descrita no Capítulo 1 (seção 1.4) seguindo o modelo de questão ampla dos estudos qualitativos: como os arquitetos de TI realizam suas atividades de arquitetura na indústria? Para tentar responder a esta questão, pessoas que realizam atividades de arquitetura em organizações de TI foram entrevistadas. O foco da pesquisa são os arquitetos de TI, mas outros papéis também foram entrevistados para que fosse possível obter um conhecimento mais amplo sobre o objeto de estudo e sobre o contexto de trabalho dos arquitetos. Isto também permitiu que fossem coletadas informações sobre atividades de arquitetura que são realizadas por papéis que não são chamados de arquitetos (o que também será descrito neste trabalho).

Assim, quatro diferentes períodos de coleta de dados foram realizados, sempre intercalados com períodos de análise de dados. No total foram conduzidas 27 entrevistas com 21 informantes diferentes em 9 organizações de TI. As entrevistas duraram entre 23 minutos e 1 hora e 27 minutos, somando em média 19 horas de entrevistas que transcritas formam mais de 380 páginas.

Na primeira etapa de coleta de dados foram realizadas 8 entrevistas em 5 empresas localizadas em Porto Alegre. Essas entrevistas focaram em aspectos que influenciam o desenvolvimento da arquitetura de software, especialmente quando a empresa trabalha com DDS, enfatizando a importância do trabalho dos arquitetos. Inicialmente, deu-se preferência a empresas que trabalhassem em projetos distribuídos, uma vez que a distância geográfica característica desse tipo de desenvolvimento intensifica os desafios da engenharia de software (Herbsleb *et al.*, 2001), especialmente aqueles relacionados à coordenação e comunicação, ambos aspectos em que a arquitetura (e os arquitetos) exercem grande influência (Grinter, 1999)(Ovaska *et al.*, 2003). O guia desta etapa de entrevistas pode ser encontrado no

Apêndice B deste trabalho e é composto por várias perguntas abordando diversos aspectos que podem influenciar a arquitetura. A Tabela 3.2 a seguir traz um resumo dos entrevistados nesta primeira etapa de coleta de dados, seus papéis e experiência (seguindo o padrão ñano,mesesò) na empresa e no papel desempenhado. Por motivos de confidencialidade as informações estão apresentadas de maneira anônima.

Tabela 3.2 Resumo dos informantes da primeira etapa de coleta de dados

Empresa	Informantes e papéis	Tempo na empresa/tempo no papel
A	Informante 1: Arquiteto técnico	7/1,6
B	Informante 2: Líder técnico	4,6/4,6
	Informante 3: Desenvolvedor	1,6/1,6
C	Informante 4: Arquiteto	4/2
	Informante 5: Arquiteto	5/3
D	Informante 6: Gerente de projetos distribuídos	6/6
E	Informante 7: Coordenador de tecnologia/Arquiteto	11/3 ï 11/6
	Informante 8: Gerente de tecnologia	6/2

As entrevistas foram transcritas e analisadas usando a ferramenta MaxQDA (Verbi Software, 2010). É importante mencionar que nesta e na etapa seguinte as entrevistas foram realizadas com o apoio do mestrando Marcelo Zílio da PUCRS, que também contribuiu para a transcrição e codificação das mesmas. A etapa de codificação aberta foi realizada, separando os dados em categorias e subcategorias. Em seguida, a etapa de codificação axial foi iniciada para compreender o que os dados significavam.

A Figura 3.3 a seguir traz o conjunto de códigos criados durante a análise dos dados nesta etapa contendo as categorias e suas subcategorias. A partir da análise dos dados desta etapa, dois aspectos foram identificados: o uso de arquitetura orientada a serviços (SOA) e a presença de diferentes tipos de arquitetos. Os aspectos referentes a SOA foram classificados dentro da subcategoria ñTecnologiaò, parte da categoria ñArquiteturaò. Este aspecto foi abordado no trabalho de dissertação do Marcelo Zílio e não será discutido neste trabalho (Pereira, 2011)(Pereira *et al.*, 2011).O código chamado de ñTipos de arquitetosò foi criado para agregar as informações sobre esse aspecto. Neste momento, foi identificado que duas empresas (A e E) possuíam diferentes tipos de arquitetos e resolveu-se investigar melhor este aspecto nas etapas seguintes de coleta de dados.

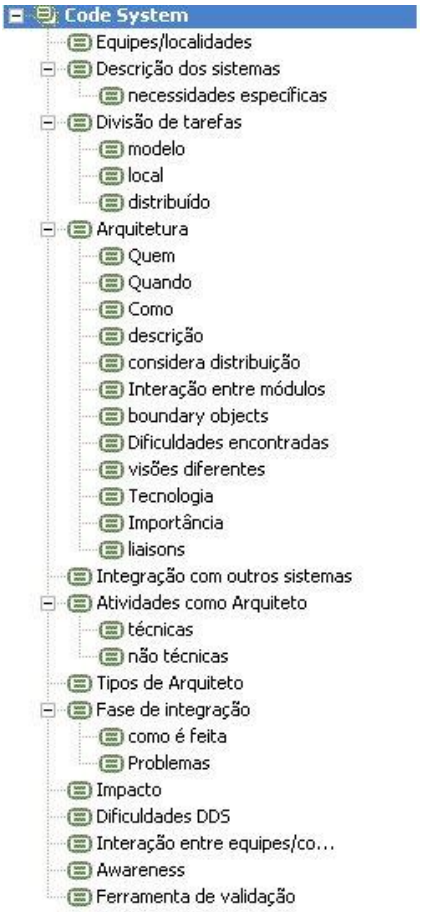

Figura 3.3 Conjunto inicial de códigos

Com base nessa análise inicial, a etapa seguinte de coleta de dados foi preparada. Nesta etapa já foi utilizado o conceito de amostragem teórica para descobrir o que acontece quando variam as condições sob as quais o fenômeno de interesse ocorre (Strauss e Corbin, 2008), testando assim algumas hipóteses (baseadas nos dois aspectos mais relevantes encontrados ao final da primeira etapa). Por exemplo, como se percebeu que existiam alguns tipos diferentes de arquitetos institucionalizados em algumas empresas, procurou-se na etapa seguinte investigar o que ocorria quando tais tipos de arquitetos específicos não eram encontrados institucionalizados nas empresas.

Assim, na segunda etapa procurou-se realizar tanto entrevistas de continuidade com informantes da primeira etapa quanto entrevistas com novos informantes. As entrevistas de continuidade focaram nos aspectos previamente identificados e em esclarecimentos sobre aspectos mencionados na primeira etapa, enquanto que as novas entrevistas tiveram um guia que mesclava perguntas gerais contidas na primeira etapa com perguntas focadas nos aspectos mais relevantes da etapa anterior. Desta forma, cinco entrevistas foram realizadas sendo três

de continuidade e dois em uma nova empresa. O guia de entrevistas desta etapa pode ser visto no Apêndice C deste trabalho[8]. A Tabela 3.3 sumariza esta segunda etapa de coleta de dados. Também nesta etapa as entrevistas foram transcritas e analisadas usando a MaxQDA: os dados foram integrados às mesmas categorias da primeira etapa e analisados em conjunto. Caso novos códigos precisassem ser adicionados, todas as entrevistas eram analisadas novamente; este processo ocorreu ao longo de toda a pesquisa. Assim, as codificações aberta e axial também foram realizadas com os dados dessa segunda etapa.

Tabela 3.3 Resumo dos informantes da segunda etapa de coleta de dados

Empresa	Tipo de entrevista	Informantes e papéis	Tempo na empresa/tempo no papel
A	continuidade	Informante 1: Arquiteto técnico	7/1,6
C	continuidade	Informante 4: Arquiteto	4/2
		Informante 5: Arquiteto	5/3
F	novas	Informante 9: Arquiteto	5/5
		Informante 10: Arquiteto	4/4

Nesta etapa, refinou-se o aspecto relacionado à presença de diferentes ñtipos de arquitetosò participando nas atividades de arquitetura. Foi identificado que todas as empresas pesquisadas possuíam divisões similares das atividades de arquitetura, mesmo quando os tipos de arquitetos não eram institucionalizados (o que será melhor explicado no capítulo seguinte). Identificou-se também a consequente necessidade de interação entre esses diferentes tipos de arquitetos e a relação entre os tipos de arquitetos e aspectos de difusão de informações, comunicação e coordenação durante o processo de desenvolvimento de sistemas de TI. Assim, a questão e o foco de pesquisa foram refinados com base nos dados obtidos, conforme previsto pela Teoria fundamentada em dados (Strauss e Corbin, 2008).

A etapa de codificação seletiva também foi iniciada neste momento. Nesta etapa de codificação, uma categoria principal é eleita ou criada para descrever a teoria através dela. Ao analisar os dados através do aspecto descrito anteriormente (os vários tipos de arquitetos), notou-se que as empresas não somente apresentam diferentes tipos de arquitetos, mas também estes estão organizados de uma forma tal que apoiam o desenvolvimento dos sistemas de software que essas empresas criam. Com isso, uma nova categoria foi criada com o nome de ñTraduçãoò representando o processo pelo qual as necessidades dos clientes são traduzidas em aspectos técnicos que podem ser compreendidos e desenvolvidos pelos desenvolvedores

[8]É importante mencionar que o Apêndice C traz apenas o guia geral, sem as perguntas específicas conduzidas para os informantes que fizeram entrevistas de continuidade, uma vez que tais perguntas eram voltadas a clarificar aspectos das entrevistas anteriores de cada um deles.

dentro do contexto (sistemas, soluções, ferramentas, etc) organizacional existente. Conforme será discutido no próximo capítulo, os diferentes tipos de arquitetos desempenham uma função importante nesse processo.

Durante a análise dos dados desta etapa, após a identificação de que todas as empresas dividiam suas atividades de arquitetura de forma similar, o trabalho de Akenine (2008) foi encontrado relatando a pesquisa da IASA que recomenda quatro tipos diferentes de arquitetos (ver seção 2.3.3). Percebeu-se com isso que a divisão sugerida pela IASA era bastante parecida com a divisão das atividades de arquitetura encontradas nas empresas. Assim, os dados foram analisados novamente para esclarecer este aspecto e o conjunto de códigos foi atualizado de acordo com o refinamento da pesquisa e da definição da categoria principal e considerando o trabalho de Akenine (2008). Por exemplo, utilizou-se parte da nomenclatura definida pela IASA citada no trabalho de Akenine. A Figura 3.4 traz o conjunto de códigos já atualizado.

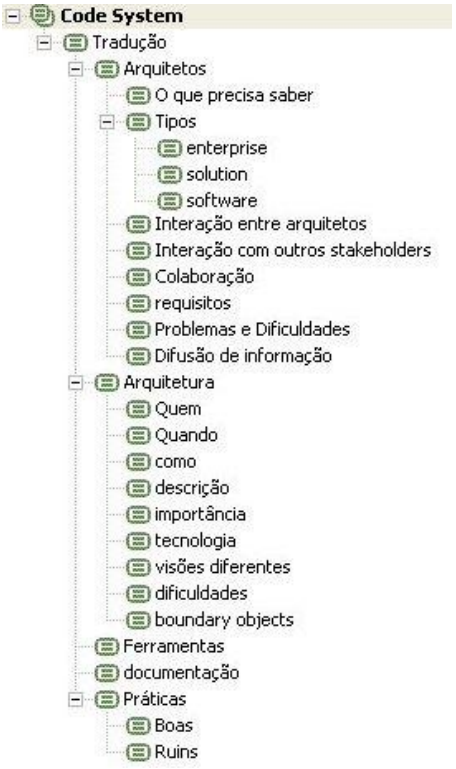

Figura 3.4 Conjunto de códigos atualizado

A Figura 3.5 a seguir traz uma imagem da ferramenta MaxQDA2 contendo um trecho de uma entrevista já categorizado de acordo com esse conjunto de códigos. A região marcada com o número 3 apresenta um trecho da transcrição de uma entrevista. Ao lado do texto é

possível ver colchetes que indicam que aquele pedaço de texto foi anexado a uma categoria (na ferramenta utilizada, é possível ver o nome da categoria posicionando o *mouse* sobre o colchete). A região 1 indica os arquivos (entrevistas) que são codificados. A região 2 mostra o conjunto de categorias criada durante a etapa de codificação aberta. Por fim, a região 4 serve para visualizar os trechos anexados a uma determinada categoria.

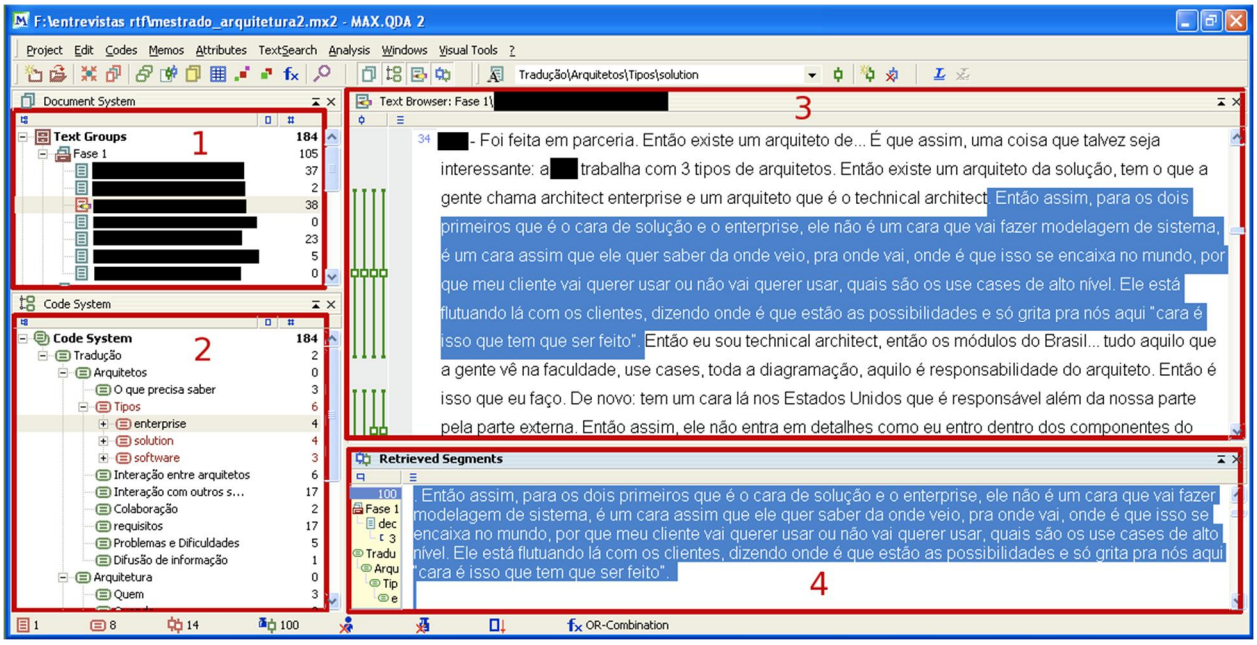

Figura 3.5 Exemplo do uso da ferramenta MaxQDA para codificação

A área 3 da Figura 3.5 é detalhada na Figura 3.6 a seguir, na qual observa-se um exemplo de codificação. Nela é possível observar um trecho de uma entrevista na janela principal e, à esquerda, os códigos anexados a partes de texto do documento. Por exemplo, a categoria chamada ñTipos de arquitetosò foi criada para classificar trechos que falem sobre diferentes tipos de arquitetos de TI que existam dentro das empresas, sejam eles chamados de arquitetos ou não, institucionalizados ou não. O trecho destacado está anexado a esta categoria e corresponde à explicação dos tipos de arquitetos que a empresa em questão possui. Essa categoria foi dividida em subcategorias para os principais tipos de arquitetos encontrados, usando a nomenclatura do estudo da IASA descrito por Akenine (2008) e o mesmo trecho destacado foi anexado a duas delas (correspondentes aos tipos de arquitetos que menciona). Além disso, é interessante identificar como é a interação entre os arquitetos, por isso uma categoria ñInteração entre arquitetosò também foi criada dentro da categoria ñTipos de arquitetosò, sendo que o mesmo trecho em questão também trata desta interação e por isso também foi classificado na categoria ñInteração entre arquitetosò.

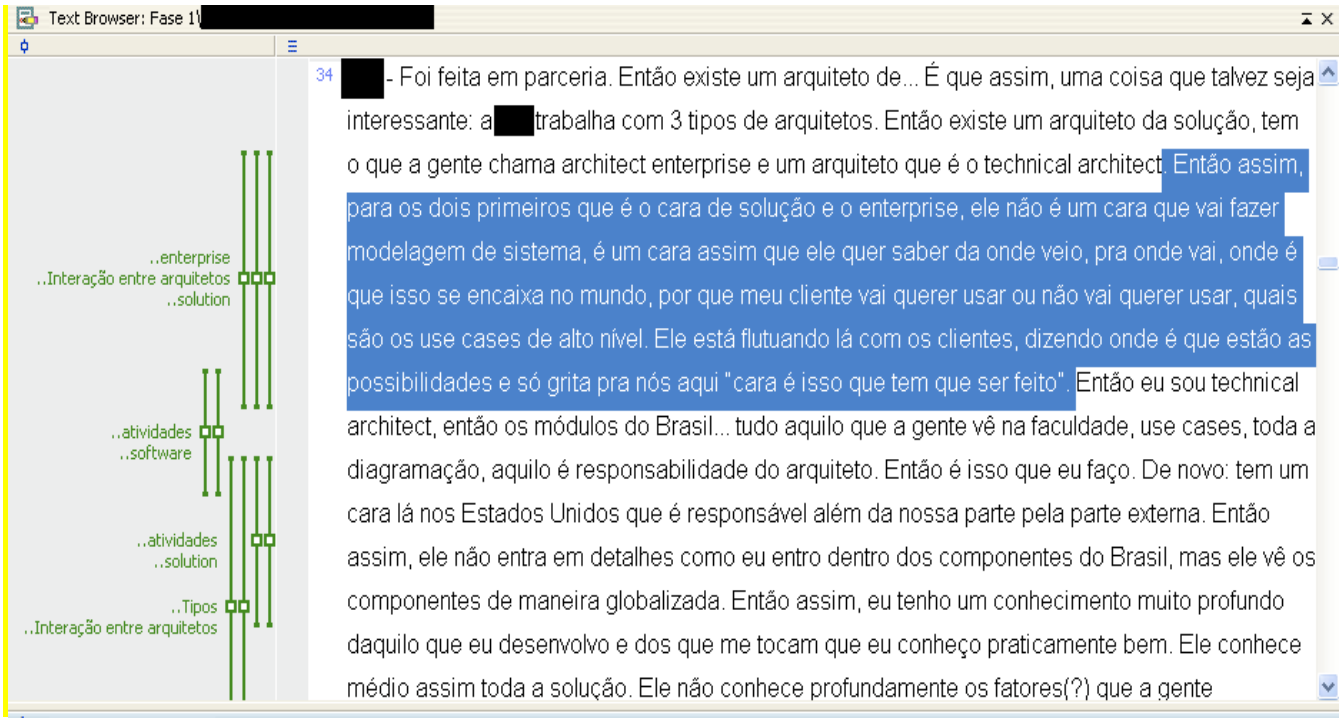

Figura 3.6 Exemplo de codificação

A análise da segunda etapa guiou a etapa seguinte de coleta de dados. Nesta, três novas entrevistas foram conduzidas em uma empresa localizada em Belém, usando novamente amostragem teórica, ou seja, escolhendo pontos de coleta de dados para expandir e/ou testar a teoria em desenvolvimento. Neste caso, escolheu-se uma empresa que não apresentava os diferentes tipos de arquitetos institucionalizados. A Tabela 3.4 a seguir resume esta etapa.

Tabela 3.4 Resumo dos informantes da terceira etapa de coleta de dados

Empresa	Tipo de entrevista	Informantes e papéis	Tempo na empresa/tempo no papel
G	novas	Informante 11: Arquiteto	2,6/2,6
		Informante 12: Arquiteto	3/3
		Informante 13: Arquiteto	2/0,8

Mais uma vez as entrevistas foram transcritas e analisadas com a MaxQDA2. Após esta etapa, a teoria já estava melhor definida de forma que o planejamento da etapa seguinte já levou em consideração aspectos de validação da mesma: de acordo com Strauss e Corbin (2008), uma forma de validar a teoria é contá-la aos informantes e pedir a eles que comentem se e como ela se ajusta aos seus casos. É claro que a teoria não vai se ajustar em todos os aspectos de todos os casos, pois se trata de uma redução dos dados, mas em um sentido mais

amplo os informantes devem ser capazes de reconhecer o que ocorre nas suas empresas na teoria proposta, percebendo-a como uma explicação razoável (Strauss e Corbin, 2008). Assim, na quarta etapa de coleta de dados tentou-se entrevistar todos informantes que forneceram informações nas etapas prévias. Além disso, para evitar que os dados fossem restritos ao ponto de vista de uma única pessoa, ou seja, que não fossem representativos da organização como um todo, tentou-se também contatar pelo menos um novo informante de cada empresa onde a pesquisa foi realizada. Esta foi outra forma de validar o trabalho aqui descrito. Finalmente, foram conduzidas entrevistas em novas empresas para verificar se a teoria também se aplicaria a elas. A Tabela 3.5 a seguir resume as entrevistas e seus informantes desta quarta etapa de coleta de dados. Assim como nos períodos anteriores, estas entrevistas foram transcritas e analisadas. Infelizmente não foi possível retornar às Empresas B e G nesta quarta etapa e nem contatar um novo informante da Empresa F.

Tabela 3.4 Resumo dos informantes da quarta etapa de coleta de dados

Empresa	Tipo de entrevista	Informantes e papéis	Tempo na empresa/tempo no papel
A	continuidade	Informante 1: Arquiteto técnico	7/1,6
	nova	Informante 14: Arquiteto técnico	-
	nova	Informante 15: Arquiteto	-
C	continuidade	Informante 4: Arquiteto	4/2
	continuidade	Informante 5: Arquiteto	5/3
	nova	Informante 16: Arquiteto	6/2
D	nova	Informante 17: Arquiteto	6/6
E	continuidade	Informante 7: Coordenador de tecnologia/Arquiteto	11/3 ï 11/6
	continuidade	Informante 8: Gerente de Tecnologia	6/2
	nova	Informante 18: Arquiteto	-
F	continuidade	Informante 9: Arquiteto	5/5
H	nova	Informantes 19 e 20: Consultores de sistemas	0,9/0,9 ï 0,4/0,4
I	nova	Informante 21: Arquiteto	4/2

Dessa forma, a Teoria fundamentada em dados forneceu uma perspectiva através da qual foi possível examinar o trabalho dos arquitetos de TI e compreender como eles influenciam e contribuem para que as necessidades dos clientes sejam traduzidas para os termos técnicos que permitem que os desenvolvedores desenvolvam o produto requerido. Para isso, aspectos importantes referentes às interações entre os arquitetos (entre si e com outros *stakeholders*) foram também examinados, bem como as ferramentas de apoio utilizadas por

eles. O próximo capítulo apresentará os resultados dos períodos de coleta e análise de dados realizados nessa pesquisa.

CAPÍTULO 4

4. Resultados

4.1. Visão geral

Neste trabalho foram realizadas entrevistas com profissionais que trabalham no contexto da arquitetura de TI na prática e, a partir das informações providas por eles, foi possível identificar algumas estratégias utilizadas na indústria com relação à arquitetura de TI. Uma delas é a existência de vários tipos de arquitetos, dentre os quais está o arquiteto de software. Tais tipos de arquiteto não se concentram apenas nas atividades relacionadas a modelos arquiteturais, linguagens de modelagem, padrões de projeto, entre outras mais comumente estudadas na academia. Eles estão envolvidos no contexto maior de TI da empresa e exercem atividades desde o momento do contrato. Exemplos destas atividades são participar da definição do contrato, definir os padrões de TI adotados pela empresa, interagir com clientes e analistas para identificação e compreensão de requisitos (especialmente não funcionais), entre outras.

Além disso, observou-se que, mesmo cada empresa apresentando a sua própria organização de tipos de arquiteto, elas acabam apresentando uma configuração similar, com várias semelhanças em relação às tarefas que eles executam. Com isso foi possível definir três grupos diferentes para as empresas de acordo com a forma com que estes tipos de arquiteto estão organizados. A partir dessa classificação, foi possível identificar como esse tipo de divisão influencia as atividades de desenvolvimento de software e o trabalho dos arquitetos.

Este capítulo apresentará estes resultados, descrevendo essa estratégia de divisão de tipos de arquitetos que as empresas adotam, classificando-as e analisando essa divisão de acordo com as atividades realizadas pelos arquitetos, para que, no capítulo seguinte, tais resultados possam ser discutidos no contexto de outros trabalhos científicos, bem como a contribuição destes trabalhos para a pesquisa em arquitetura de software.

4.2. Tipos de arquitetos na prática

Como discutido no Capítulo 1, alguns órgãos como a IASA e o The Open Group sugerem que os arquitetos recebam um treinamento base comum e que após isso se aprofundem em especializações (para a IASA) ou disciplinas (para a certificação ITAC do The Open Group).

Além disso, alguns estudos (Akenine, 2008)(Unde, 2008) realizados juntamente com empresas também sugerem a existência de uma divisão das atividades de arquitetura de TI em vários papéis. De fato, como detalhado na seção 2.3.3, Akenine (2008) descreve um estudo da IASA que, ao final, sugere uma padronização de 4 papéis diferentes de arquiteto de TI: arquiteto *enterprise*, arquiteto de negócios, arquiteto de soluções e arquiteto de software.

Os resultados deste trabalho estão alinhados com essas abordagens: eles sugerem que as organizações observadas possuem diferentes tipos de arquiteto. Entretanto, os resultados aqui apresentados vão além disso: eles indicam que mesmo quando as empresas observadas não possuem esses tipos de arquitetos formalmente definidos e institucionalizados, ainda assim é possível identificar uma divisão nas atividades de arquitetura, que pode ser feita pelos próprios arquitetos e também envolver atores que não são chamados de arquitetos. Mais que isso: tal divisão geralmente apresenta-se de forma similar. Por exemplo, a Empresa A possui 3 tipos de arquitetos muito bem definidos e formalizados, enquanto a Empresa F possui uma equipe de 4 arquitetos (chamados simplesmente de arquitetos de software) que não possuem divisão de atividades pré-definida, mas que as dividem mesmo assim por iniciativa própria. Note que as Empresas A e F do exemplo do parágrafo anterior representam casos extremos: uma com vários tipos de arquitetos formalizados e a outra sem nenhuma especificidade formalizada. Entretanto, algumas empresas como a Empresa C se encontram em um ponto intermediário. Nestas não há uma formalização como na Empresa A e os arquitetos são chamados simplesmente arquitetos de software como na Empresa F, porém ao se analisar sua estrutura e suas atividades de arquitetura, percebe-se que existem outros papéis, que não são chamados arquitetos, que exercem atividades de arquitetura, dividindo assim tais atividades com os arquitetos da empresa. E, mesmo com essa diferença de formalização, essa divisão também recai em uma estrutura similar para as três empresas usadas como exemplo.

Apesar das diferenças entre essas três empresas, observa-se que a divisão de atividades de arquitetura que existe nelas é similar. Na verdade, comparando os resultados com os papéis de arquitetos sugeridos pela IASA conforme descrito por Akenine (2008), é possível identificar que a divisão de atividades encontrada nas organizações pesquisadas neste trabalho assemelha-se com a estrutura representada na Figura 4.1 a seguir. Nesta figura, os tipos de arquiteto mais comuns estão representados juntamente com os *stakeholders* mais comuns com os quais eles costumam interagir. Note que os *stakeholders* que não são da mesma empresa que os arquitetos estão representados com a cor preta, diferente dos demais.

Figura 4.1 Estrutura genérica dos tipos de arquiteto e suas principais interações

Na Figura 4.1 os três principais tipos de arquiteto, encontrados durante a pesquisa estão representados: o arquiteto *enterprise*, o arquiteto de soluções e o arquiteto de software. Estes três tipos de arquitetos foram inspirados na divisão de papéis da IASA (descrita na seção 2.3.3), pois suas atividades são similares. Assim, durante a pesquisa procurou-se identificar nas empresas atores que realizassem as atividades que caracterizam cada tipo de arquiteto, mesmo que estes não ocupassem cargos com o título de arquiteto. Essa identificação foi feita através das seguintes analogias: o arquiteto *enterprise* será comparado a atores cujas atividades de arquitetura correspondem ao fechamento do acordo com o cliente, relacionamento com a gerência e visão de negócio; o arquiteto de soluções será comparado a atores cujas atividades de arquitetura estejam voltadas para a definição da visão geral da arquitetura, principais componentes e seus relacionamentos, sem entrar em detalhes de código ou definições detalhadas; e por fim, o arquiteto de software será comparado com atores que realizem atividades relacionadas à arquitetura que envolvam o refinamento da visão geral da arquitetura, a definição de padrões que serão usados no desenvolvimento e que também mantenham contato mais próximo com o time de desenvolvimento.

Assim, a Figura 4.1 apresenta uma estrutura genérica das divisões encontradas e foi criada para representar os resultados desta dissertação. É preciso ressaltar que as configurações das empresas são diferentes e a divisão das atividades de arquitetura é feita de acordo com os interesses e necessidades de cada empresa. Sendo assim, a Figura 4.1 é uma espécie de generalização dos esquemas encontrados nas empresas pesquisadas, sugerindo uma tendência dentro dessas organizações.

É possível identificar em todas as organizações pesquisadas um denominador comum: sempre existem atores realizando as atividades do arquiteto de soluções e do arquiteto de software, mas a forma que isto é organizado é diferente em cada empresa pesquisada. Por sua vez, as atividades do arquiteto *enterprise* apareceram concentradas em um papel específico apenas em duas empresas (A e E), sendo que uma possui este tipo de arquiteto bem definido e formalizado (Empresa A), enquanto a outra (Empresa E) apresenta uma configuração mais peculiar, mas onde ainda assim é possível identificar tais atividades bem destacadas (as configurações de cada empresa serão descritas na seção seguinte). Nas empresas em que este tipo de arquiteto não aparece, algumas de suas atividades podem ser realizadas pelo ator que executa as atividades do arquiteto de soluções, ou mesmo pode acontecer de tais atividades de arquitetura não serem realizadas na empresa, por exemplo, pode acontecer de que ninguém que exerça atividades de arquitetura participe da definição do contrato junto ao cliente.

Durante a pesquisa o papel de arquiteto de negócios (segundo a IASA) não foi identificado de maneira bem definida em nenhuma das organizações pesquisadas. Entretanto, observou-se que, quando existe um arquiteto *enterprise* (ou similar) na empresa, este ator também realiza algumas das atividades de um arquiteto de negócio. Por exemplo, na Empresa A, o arquiteto *enterprise* também trabalha para criar tendências de mercado, ou seja, criar necessidades que o negócio da empresa pode suprir e que os clientes nem haviam pensado nelas antes; uma atividade mais associada ao arquiteto de negócios.

Na Figura 4.1 também é possível observar os principais grupos de *stakeholders* com quem os diferentes de tipos de arquitetos interagem. É importante ressaltar que o que é apresentado na figura não representa uma hierarquia; isto é, um tipo de arquiteto não necessariamente responde ao outro. Ela também não representa um ñmodelo em cascataò, onde um arquiteto só começará suas atividades após o outro; muito pelo contrário, os vários arquitetos trabalham em conjunto. O que a Figura 4.1 representa, e que foi observado durante a pesquisa, é que essa configuração representa o fluxo de informações entre os arquitetos. Em outras palavras: cada tipo de arquiteto recebe informações de diferentes grupos de *stakeholders*, realiza seu trabalho acrescentando suas próprias informações e, depois disso, passa adiante um tipo diferente de informação. Esse fluxo de informações está representado através das setas, mostrando as interações entre os arquitetos e entre eles e outros *stakeholders*. Assim, os diferentes tipos de arquitetos trabalham tanto como agregadores quanto como difusores de informação dentro dos projetos. São *agregadores* por receberem

informações diferentes, vindas de diferentes tipos de *stakeholders* e em formatos diferentes, as quais precisam agregar para realizar seu trabalho atendendo às necessidades provenientes destes *stakeholders*. Eles são *difusores* porque, através de suas interações com diferentes *stakeholders*, repassam as informações recebidas já modificadas com o seu trabalho para eles. Esse aspecto está relacionado com a recomendação de Grinter (1999) de os arquitetos (de software no caso dela, mas aqui estendida para os arquitetos de TI) atuarem como *boundary spanners*, que foi descrita na seção 2.4.1.

Com base nas observações citadas nesta seção, pôde-se observar que as empresas pesquisadas podem ser divididas em 3 grupos diferentes que serão descritos a seguir. É importante destacar que não foi possível retornar após a primeira etapa na Empresa B e por isso não foi possível obter essas informações dessa organização. Não foi possível também identificar essas informações na Empresa H, uma vez que apenas uma entrevista foi realizada nesta organização e ela não foi conclusiva, além do fato de a filial desta empresa onde a pesquisa foi realizada ainda estar se instalando no Brasil. Por isto, estas duas empresas não estão descritas a seguir. O resultado da classificação de todas as outras é descrito a seguir.

4.2.1. Classificação

De acordo com a descrição da configuração dos tipos de arquitetos e das atividades de arquitetura que eles realizam em cada empresa, é possível identificar que nas empresas observadas no mínimo existe uma diferenciação em dois tipos: um que realizará as atividades de arquiteto de soluções e outro que atuará como arquiteto de software (como na Figura 4.1). Observando o trabalho que estes dois tipos de arquitetos realizam percebe-se que a disciplina de arquitetura de software na academia normalmente lida com as atividades referentes a esses dois tipos. Entretanto, enquanto a academia normalmente credita estas atividades a um único tipo de arquiteto, nas empresas pesquisadas as mesmas atividades estão divididas em dois tipos diferentes. Além disso, as atividades de arquiteto *enterprise* somente começam a ser definidas quando os arquitetos de soluções e de software são melhor identificados, sendo que o tipo de arquiteto *enterprise* só foi encontrado nas empresas que apresentam os outros dois bem definidos. Com isso, é possível perceber os dois extremos das configurações das empresas. Primeiro, a Empresa A que apresenta os três tipos de arquiteto institucionalizados com suas interações bem definidas. O segundo extremo é o das Empresas F e G, que não possuem diferentes tipos de arquitetos formalizados, mas que mesmo assim eles dividem as atividades de arquitetura entre si. No meio disso estão as empresas onde existem outros

atores, além dos arquitetos, que realizam atividades de arquitetura, mas estes não recebem o nome de arquiteto. Portanto, as diferenças entre a organização dos tipos de arquiteto das empresas pesquisadas podem ser classificadas em três grupos diferentes de acordo com a definição dos tipos nelas encontrados. São eles: ñtipos de arquitetos definidosò, ñtipos de arquitetos parcialmente definidosò e ñtipos de arquitetos não definidosò. Esses grupos são resumidos na Tabela 4.1 a seguir.

Tabela 4.1 Mapeamento dos papéis de arquiteto da IASA e dos tipos de arquitetos encontrados nas empresas pesquisadas

Grupo	Empresa	Papéis IASA	Nome do papel
Tipos de arquitetos definidos	A	Arquiteto *enterprise*	Arquiteto *enterprise*
		Arquiteto de soluções	Arquiteto de soluções
		Arquiteto de software	Arquiteto técnico
	E	Arquiteto *enterprise*	Equipe funcional (responsável funcional e equipes de tecnologia) + equipe técnica (responsável técnico, desenvolvedores e *delivery*de serviços)
		Arquiteto de soluções	Responsável técnico
		Arquiteto de software	Líder técnico
Tipos de arquitetos parcialmente definidos	C	Arquiteto *enterprise*	-
		Arquiteto de soluções	Arquiteto de software
		Arquiteto de software	Projetista
	D	Arquiteto *enterprise*	-
		Arquiteto de soluções	Arquiteto de software
		Arquiteto de software	Desenvolvedor sênior
	I	Arquiteto *enterprise*	-
		Arquiteto de soluções	Arquiteto de software
		Arquiteto de software	Líder técnico
Tipos de arquitetos não definidos	F	Arquiteto *enterprise*	-
		Arquiteto de soluções	Arquiteto de software
		Arquiteto de software	Arquiteto de software
	G	Arquiteto *enterprise*	-
		Arquiteto de soluções	Arquiteto de software
		Arquiteto de software	Arquiteto de software

A Tabela 4.1 apresenta os três grupos identificados na primeira coluna e como cada empresa é mapeada para cada um destes grupos na segunda coluna. Na coluna seguinte, ela traz os três tipos de arquitetos representados na Figura 4.1 (inspirados nos papéis sugeridos pela IASA) e encontrados durante a pesquisa de acordo com a analogia descrita na seção anterior. Por fim, na última coluna são apresentados os nomes que cada empresa atribui aos tipos de arquiteto que realizam as atividades de cada arquiteto do esquema genérico da Figura 4.1 que aparece na segunda coluna. Por exemplo, a Empresa C está categorizada no grupo de ñtipos parcialmente definidosò (coluna 1). Esta empresa não possui um tipo de arquiteto equivalente ao de arquiteto *enterprise* (coluna 3), o que é representado pelo traço na coluna 4 (o que não significa que tais atividades são completamente inexistentes na empresa, elas são em menor número e não são destacadas das outras atividades e nem concentradas em atores específicos). Já o tipo arquiteto de soluções (coluna 3) é chamado nesta empresa de arquiteto de software (coluna 4), enquanto o tipo arquiteto de software (coluna 3) é chamado de projetista na Empresa C (coluna 4). É importante destacar que a configuração de algumas empresas apresenta apenas atores chamados de arquitetos de software, mas estes acabam por exercer papéis de tipos de arquitetos diferentes em seus projetos, conforme será descrito nas seções seguintes. Assim, a Tabela 4.1 resume a classificação das empresas nesses três grupos, os quais serão descritos a seguir.

4.2.1.1. Tipos de arquitetos definidos

Neste grupo de empresas, os tipos de arquiteto *enterprise*, de soluções e de software são claramente diferenciados. Eles podem receber nomes diferentes e também ser representados por mais de um ator (vide Tabela 4.1), mas realizam atividades correlatas às definidas na analogia da seção 4.2. Apenas neste grupo as atividades de arquiteto *enterprise* foram identificadas de forma bem definida. Além disso, nas empresas deste grupo também é possível identificar algumas atividades típicas do arquiteto de negócio sugerido pela IASA, apesar de este papel não ter sido encontrado em nenhuma das empresas pesquisadas. Tais atividades são realizadas pelo tipo de arquiteto que realiza as atividades do arquiteto *enterprise*.

As Empresas A e E estão neste grupo. Ambas são organizações multinacionais e trabalham com desenvolvimento distribuído de software em vários países. A seguir, a instanciação do esquema da Figura 4.1 nestas empresas será descrita.

4.2.1.1.1. Empresa A

A Empresa A é a que apresenta a configuração mais parecida com a representação genérica da Figura 4.1, possuindo os três papéis bem definidos que nela são chamados de arquiteto *enterprise*, arquiteto de soluções e arquiteto técnico. Os dois primeiros utilizam a mesma nomenclatura da Figura 4.1, enquanto o último, apesar de ser chamado por um nome diferente, equivale ao arquiteto de software da mesma.

Segundo os informantes desta empresa, o arquiteto *enterprise* é aquele encarregado de apoiar o negócio da empresa. Este arquiteto está em um nível mais gerencial e, assim como descrito por Unde (2008) e Akenine (2008) é responsável pela estratégia de TI da empresa como um todo. Na Empresa A ele é o encarregado de observar as tendências de mercado e a concorrência e identificar (ou criar) novas necessidades e/ou produtos. A seguir é possível ver alguns trechos de entrevistas que descrevem o papel deste arquiteto.

> **ñ***Um cara que está acostumado com infraestrutura de enterprise, de marca de mid-porte, tem uma visão de aplicação. O cara está num alto nível assim. Então, esse é arquiteto enterprise. Ele trabalha pra fazer o fechamento de negócio com o time em si. (...) Ele é um cara que tem conhecimento da indústria e vai ajudar a fazer uma proposta pra fazer o fechamento do acordo. E é nesse nível que é esse trabalho, é bem alto. Mas é um cara que tem um conhecimento muito largo e não necessariamente muito profundo, porque o conhecimento dele é muito grande pra conseguir aprofundar. (...) Ele começa a se preocupar mais na amplitude de conhecimento que na profundidade de conhecimento***ò** ï Informante 1.

> **ñ***Então esse cara* [arquiteto enterprise] *ele não só captura as necessidades dos clientes, mas ele também aposta em coisas que vão criar as necessidades nos clientes. (...) O enterprise architect está nesse ponto, ele também está trabalhando pra criar necessidades no mercado, porque isso movimenta o mercado.***ò** ï Informante 1.

Assim, o arquiteto *enterprise* é quem, juntamente com o gerente da unidade de negócio e alguém da área de marketing, define a ideia do produto que a empresa produzirá e trabalhará para fechar o contrato com o cliente. A partir dessa ideia do produto saem as

iniciativas de trabalho: a ideia proposta por ele pode ser desmembrada em vários projetos, várias soluções. E cada solução será responsabilidade de um arquiteto de soluções.

Este segundo tipo de arquiteto é o responsável pela visão geral da arquitetura do projeto em que trabalha. Ele não entra em detalhes mais técnicos, como modelagem de sistema; ele se detém na visão em alto nível do todo. Na Empresa A essa visão geral da arquitetura engloba os componentes que serão desenvolvidos por todas as localidades envolvidas, uma vez que esta empresa trabalha com DDS. A seguir algumas descrições deste tipo de arquiteto feitas pelo Informante 1.

> ***ñ****O arquiteto de solução está muito próximo do cliente. Então assim, normalmente um arquiteto de solução é um cara que vive no desenvolvimento dentro do cliente, ele não vive na Empresa A. Por exemplo, a Empresa A tem uma conta com um cliente grande, então esse solution architect, é quem representa a solução da Empresa A dentro do cliente.****ò*** ï Informante 1.

> ***ñ****Não é um cara que vai fazer modelagem de sistema, é um cara que quer saber de onde veio, pra onde vai, onde é que isso se encaixa no mundo, por que meu cliente vai querer usar ou não vai querer usar, quais são os casos de uso de alto nível****ò*** ï Informante 1.

Uma vez criada essa solução geral, ela ainda pode ser dividida em diferentes áreas, onde cada uma será de responsabilidade de um arquiteto técnico (equivalente ao arquiteto de software da Figura 4.1). Esse arquiteto é quem irá refinar a visão geral desenvolvida pelo arquiteto de soluções. Ele sim lida com diagramação de forma a descrever os requisitos que serão desenvolvidos no módulo que coube à sua equipe, como exemplificado no trecho a seguir.

> ***ñ****O technical achitect*[equivalente ao arquiteto de software], *que é o cara que está mais próximo do componente, ele trabalha com diagramação básica de artefatos, então casos de usos ou então documentos de design de alto nível, elaboração de ferramentas que dão suporte a gerência de requisitos, então é nisso que a gente está mais focado****ò*** ï Informante 1, Empresa A.

Resumindo, o arquiteto *enterprise* é aquele encarregado de apoiar o negócio da empresa, no sentido de garantir a adoção de padrões na organização e soluções com as quais a empresa está acostumada a trabalhar. O arquiteto de soluções é aquele responsável pela visão geral da arquitetura do software e, como a Empresa A trabalha com projetos distribuídos, pela visão geral do que será desenvolvido por todas as localidades, vendo os componentes de forma globalizada, enquanto o arquiteto técnico entra em detalhes nos componentes que cabem à sua equipe em sua localidade. Dessa forma, a localização de cada arquiteto dessa empresa também influi em sua estrutura: o arquiteto *enterprise* está sempre na matriz da empresa nos Estados Unidos, o arquiteto de soluções está sempre junto ao cliente e cada filial de desenvolvimento possui arquitetos técnicos que atuam próximos a suas equipes de desenvolvedores nos módulos correspondentes às suas localidades.

4.2.1.1.2. Empresa E

Na Empresa E tem-se uma configuração mais peculiar. Essa empresa é dividida em várias equipes de tecnologia representando as várias áreas dos produtos que a empresa oferece. Por exemplo: equipe de e-mail, equipe de atendimento, de arte etc. A maior parte das atividades dos projetos é realizada por equipes multidisciplinares, envolvendo pessoas de várias dessas áreas, pessoas das outras filiais, vários tipos de arquitetos, analistas e clientes (principalmente nas etapas iniciais do projeto), de forma que estes *stakeholders* juntos complementam as informações uns dos outros. Isso é ilustrado no trecho de entrevista apresentado a seguir.

> *ñNão existe um passo áarquitetura de softwareâ. Pelo menos na nossa metodologia de desenvolvimento. Não que não exista esse papel, mas não é hierárquico: a área de produtos define os requisitos, gera-se o escopo, o arquiteto avalia e passa para desenvolvimento. Não. Quando a gente tem a lista de requisitos, o arquiteto é uma peça, assim como o time de desenvolvimento, assim como o service delivery; ele não está acima nem entre, ele está junto. Papéis complementares. Eu ainda defendo muito essa ideia de complemento e não uma serializaçãoò* ï Informante 7, Empresa E.

Dessa forma, as atividades de arquitetura geralmente são realizadas por equipes multidisciplinares. Explicando melhor: quando um cliente (a área de produtos da empresa, uma vez que o produto base desta empresa é um portal de internet que oferece vários outros serviços) traz uma requisição para a empresa, há um grupo de pessoas que vai interagir com esse cliente e entender o projeto sob o ponto de vista do cliente. Neste grupo, está o papel que eles chamam de responsável funcional, que é quem responderá pela parte funcional do projeto. Esse grupo levanta essas informações para passá-las para uma equipe que irá efetivamente trabalhar no projeto. Essa equipe geralmente é formada por um arquiteto (eleito como o responsável técnico do projeto), a equipe de *delivery* de serviços e desenvolvedores. Eles vão analisar esse projeto e realizar reuniões de arquitetura, onde serão definidos os componentes principais e as linhas gerais de como o projeto será desenvolvido, ou seja, a visão geral da arquitetura. Após isso, o responsável técnico trabalhará com outros arquitetos e com os líderes técnicos que ficarão responsáveis por áreas específicas do desenvolvimento do produto. No trecho de entrevista a seguir, o Informante 7 descreve este ñprocessoò.

> ñ*Normalmente dá pra interpretar que são 3 estágios.(...) Tem um grupo de pessoas que interage com o cliente, que entende o projeto do ponto de vista do cliente e consegue traduzir isso pra uma equipe mais de baixo nível que efetivamente vai trabalhar no projeto. Eu vejo o arquiteto de software mais do lado da parte de desenvolvimento, mas sempre ouvindo o cliente também. Porque tem coisas que o cliente fala que não são requisitos mas que de alguma maneira podem influenciar o desenvolvimento. A gente implantou uma metodologia de trabalho esse ano que cria 3 figuras em qualquer projeto. O responsável funcional pelo projeto, geralmente é a área de produtos, é a área demandante do produto; um líder de projetos, que vai fazer com que o projeto ande no caminho interno e bem estruturado; e um responsável técnico pelo projeto, que é o cara que tecnicamente responde por todas as questões do projeto. Este cara técnico, é o cara que vai na área de produtos entendendo o que precisa ser feito e entende quais pessoas envolver dentro do projeto. Eu hoje estou num projeto (...) como responsável técnico. Comigo eu tenho, o arquiteto de uma área especifica de autenticação, eu tenho o líder técnico de uma outra área que vai construir esse produto, eu tenho outro líder técnico que cuida de uma parte X e eu também trabalho com consultoria em outra área especifica. Só fazendo analogia com o Produto E: o arquiteto vai cuidar da integração do sistema da*

Empresa E, o líder técnico é o cara que somente vai construir o Produto E e o meu consultor é o que vai garantir a qualidade do som do produto [Produto E é um produto que envolve som]. *(...) Mas eu também estou envolvido em todas as decisões de arquitetura de todas as áreas* ï Informante 7, Empresa E.

Assim, fazendo a analogia com o esquema genérico da Figura 4.1, pode-se comparar o trabalho inicial das duas equipes da Empresa E (equipe do responsável funcional + equipe do responsável técnico) com as atividades realizadas pelo arquiteto *enterprise*. Por exemplo, essas duas equipes se reúnem com o cliente no início do projeto para definir o que vai ser o produto e fechar o contrato. Também é função delas neste momento identificar como este novo produto vai ser enquadrado nos padrões da empresa. Além disso, essas equipes tem grande interação com a gerência da empresa também neste momento. E mais importante: todas essas atividades consideram a arquitetura, por exemplo, essas equipes trazem aspectos relacionados à arquitetura para a definição do produto que será desenvolvido, o que fica definido no contrato. Por estes motivos essa empresa foi classificada neste grupo, visto que o que foi investigado nas empresas foram as atividades relacionadas à arquitetura, não importando se elas são realizadas por um único ator ou uma equipe deles.

Por sua vez, o responsável técnico (também chamado de arquiteto de sistemas na empresa) é um arquiteto que define, reunido com sua equipe também multidisciplinar, como será a visão geral da arquitetura e também responde por ela, não se preocupando com especificações ou com todos os aspectos de diagramação. Este ator pode ser relacionado com o arquiteto de soluções. O responsável técnico pode também delegar módulos específicos para outros arquitetos da empresa, que por sua vez também desempenham atividades de arquiteto de soluções para aquele módulo específico. Os trechos de entrevistas a seguir descrevem as atividades deste arquiteto e como ele se relaciona com a equipe multidisciplinar.

ñ*Como arquiteto*[de soluções] *o meu papel (...) é determinar como as coisas devem ser feitas, olhar pras coisas que estão sendo propostas e ver se elas são viáveis, fazer a transferência que a gente tem em encontrar pontos de falha. Então é encontrar os gargalos e determinar a melhor forma de fazer as coisas pras equipes de desenvolvimento. (...) A gente resolve o problema, então a gente especifica uma solução. Esses são os principais*

*focos. Não que eu vá fazer a especificação, mas eu ajudo a determinar o que é pra especificar e alguém vai lá e especifica***ò**ï Informante 7, Empresa E.

ñ*A gente trabalha muito com essa parte de ter o que a gente chama de system architect aqui na verdade. (...) Sempre tem uma pessoa* [arquiteto de sistemas] *que é o responsável técnico do projeto. Ele define a arquitetura***ò** ï Informante 8, Empresa E.

ñ*Então a gente pega esses 3 perfis: service delivery, a equipe de desenvolvimento e o arquiteto. Pega os requisitos, coloca no quadro e começa a desenhar as caixinhas, como elas se relacionam, identifica quais os componentes vão ser reutilizados, quais vão ser construídos. E os que vão ser construídos a gente detalha dentro dele como é que vai funcionar. A gente não chega no ponto de definir código ou arquivos que vão ser construídos. Mas a gente desenha geralmente a espinha dorsal do programa, coloca ali como as relações com outros sistemas vão acontecer***ò** ï Informante 7, Empresa E.

Por fim, após essa visão geral da arquitetura ser criada ela será desenvolvida. Na equipe dos desenvolvedores sempre há um líder técnico (que geralmente é um desenvolvedor sênior) que irá gerenciar o desenvolvimento e refinar a visão geral repassada pelo responsável técnico e desenhada durante as reuniões multidisciplinares.

4.2.1.2. Tipos de arquitetos parcialmente definidos

Este grupo abrange as Empresas C, D e I. Estas empresas possuem mais de 100 empregados e desenvolvem mais de 50 projetos por ano, o que inclui desenvolvimento de projetos em paralelo. Elas também possuem alguma experiência com times e/ou clientes distribuídos, mas ou são poucos projetos ou tratam-se de projetos piloto para testar este tipo de desenvolvimento. Ou seja, DDS não é algo consolidado nestas empresas.

Nas empresas deste grupo, o papel dos arquitetos *enterprise* não foi claramente encontrado. É possível identificar algumas de suas atividades, mas sem nenhum ator dedicado inteiramente a elas. No entanto, elas são realizadas na empresa pelo mesmo ator que realiza as atividades de arquiteto de soluções. Portanto, as empresas deste grupo não apresentam nenhum papel (ou grupo), arquiteto ou não, que seja correlato ao arquiteto *enterprise*.

Por sua vez, o papel de arquiteto de soluções é definido pela organização; ele é institucionalizado. Este papel recebe o nome de arquiteto de software nas três empresas (vide Tabela 4.1), sendo que em todas estas, estes arquitetos realizam fundamentalmente as mesmas atividades básicas do arquiteto de soluções da Figura 4.1, em particular aquelas relacionadas com a definição da visão geral da arquitetura.

Finalmente, o papel correlato ao arquiteto de software da Figura 4.1 é encontrado em todas as empresas pesquisadas classificadas neste grupo, mas não com este título: eles recebem diferentes nomes em cada empresa (vide Tabela 4.1), mas realizam fundamentalmente as mesmas atividades, sendo encarregados do refinamento da visão geral da arquitetura proposto pelo arquiteto de soluções (ou similar). Frequentemente, este papel é ocupado por um desenvolvedor experiente que adquire uma visão mais abrangente da solução, que consegue perceber a solução como um todo, sem estar restrito ao desenvolvimento de módulos específicos.

A seguir, a instanciação do esquema da Figura 4.1 nas Empresas C, D e I é descrita.

4.2.1.2.1. Empresa C

Nesta empresa existe apenas um tipo de arquiteto chamado de arquiteto de software que realiza as atividades relativas ao arquiteto de soluções da Figura 4.1. Além disso, foi possível identificar outro ator, que não é chamado de arquiteto, mas que exerce as atividades de arquitetura correspondentes ao arquiteto de software da Figura 4.1: o projetista.

Quando um projeto inicia na Empresa C, uma equipe é criada para discutir o mesmo. Ela é formada pelos clientes, analistas de negócio, arquiteto (de soluções) e pessoas responsáveis pela infraestrutura. Com isso, o arquiteto (de soluções) começa seu trabalho antes mesmo de os analistas funcionais (analistas de requisitos) iniciarem o seu. Essa equipe é quem irá fechar o contrato que definirá o que a empresa irá desenvolver. Portanto, o arquiteto da Empresa C (similar ao arquiteto de soluções) desempenha as atividades de um arquiteto *enterprise* relacionadas com a definição do contrato, que acontece nessa etapa inicial. A partir do resultado gerado por essa equipe inicial o arquiteto de software da Empresa C passa a agir fundamentalmente como um arquiteto de soluções: desenvolvendo a visão geral da arquitetura. Portanto, este único tipo de arquiteto da Empresa C acumula atividades de dois tipos de arquitetos da Figura 4.1. Os informantes da Empresa C definem as atividades deste arquiteto como nos seguintes trechos:

ñ*O arquiteto tem uma função bem separada disso tudo: ele pega a ideia geral do projeto, o que vai ser o projeto. Ele define aquilo, ele entende qual é o nível de investimento que vai ser feito no projeto, e isso às vezes não depende do analista em si, depende da relação comercial. A gente já recebe algumas informações lá na hora que está fazendo o contrato: esses caras estão adotando com a Microsoft a plataforma tal e eles já estão comprando isso e aquilo e já estão montando um ambiente... Quer dizer a gente já sabe algumas informações e o analista* [funcional] *ainda nem começou o trabalho dele, mas a gente já sabe. Então é mais com essas informações que a gente trabalha. Então esse cara monta, com base nessas ideias todas, como é que vai ser o conceito do sistema em si que vai ser desenvolvido e projeta aquilo que eu falei: a visão do conceito, depois uma visão lógica e tal. Ele não se prende ainda no desenvolvimento em si. Ele define como é que as peças vão se ligar, qual é o produto mais adequado pra cada situação****ò*** ï Informante 4, Empresa C.

ñ*O arquiteto, via de regra, vai determinar os guidelines, as caixas maiores****ò*** ï Informante 16, Empresa C.

É importante ressaltar também o trabalho dos analistas funcionais. Eles começam suas atividades após o momento inicial de fechamento do contrato. A partir daí são eles que concentram a comunicação com o cliente; o arquiteto e/ou projetista passa a ter pouco ou nenhum contato com os mesmos. Portanto eles se tornam uma peça ainda mais importante para o fluxo de informações.

O projetista é o encarregado de refinar a arquitetura proposta pelo arquiteto de software (similar ao arquiteto de soluções). Enquanto o arquiteto de software se preocupa com a forma como os componentes principais são organizados, o projetista irá detalhá-los, instanciando os requisitos para a arquitetura proposta, como exemplificado a seguir:

ñ*O arquiteto* [correlato ao arquiteto de soluções] *faz toda essa parte de revisões funcionais, todo esse trabalho antes, definir a arquitetura física, lógica e tudo mais, e o projetista se preocupa mais em pegar a arquitetura já pronta, pacotes, componentes centrais, e projetar a utilização daquilo pra usar os requisitos. É mais assim: fazer o link*

*entre o que foi definido na arquitetura e o que na prática vai ser usado.***ò** ï Informante 5, Empresa C.

ñ*O papel* [do projetista] *dentro do projeto, na nossa leitura, ele projeta só os componentes, ou seja, ele pega a arquitetura que já está instanciada, já está criada, e começa instanciar componentes da arquitetura. Então basicamente é modelar o sistema dentro da arquitetura que já está sendo proposta, é manter o padrão do que foi definido na arquitetura***ò** ï Informante 5, Empresa C.

Este papel de projetista é quem tem maior contato com os desenvolvedores. É importante ressaltar que o projetista é um papel e não um cargo, portanto ele pode ser desempenhado tanto por um desenvolvedor sênior quanto por um arquiteto, dependendo das necessidades do projeto. O fato é que sempre há alguém ocupado com as tarefas de projetista, que são correspondentes às tarefas do arquiteto de software da Figura 4.1.

4.2.1.2.2. Empresa D

A Empresa D, assim como a Empresa C, também apresenta apenas atores que recebem o título de arquiteto de software. Entretanto, este arquiteto de software na verdade é similar ao arquiteto de soluções da Figura 4.1 enquanto as atividades do arquiteto de software da Figura 4.1 são realizadas na Empresa D por um desenvolvedor sênior.

Aqui novamente no início de um projeto forma-se uma equipe para interagir com os clientes e definir o que vai ser feito no projeto. Essa equipe é formada basicamente por analistas de requisitos e o arquiteto de software (similar ao arquiteto de soluções). Portanto, também na Empresa D o arquiteto (similar ao arquiteto de soluções) trabalha no início do projeto para a definição do contrato, realizando assim esta parte das atividades de um arquiteto *enterprise*. Além deste trabalho de pré-venda, o arquiteto também trabalha bem próximo aos analistas no levantamento de requisitos, porém com o foco nos requisitos não funcionais. Após essa etapa de pré-venda esse arquiteto irá definir a visão geral da arquitetura. Suas atividades são descritas pelo Informante 17 desta empresa no trecho a seguir.

> ***ñ****Como arquiteto eu trabalho bastante voltado a requisitos, principalmente requisitos não funcionais. Então, por exemplo, a primeira coisa quê eu foco pra ter o entendimento do que eu devo fazer numa nova atribuição é perguntar qual a carga que o sistema vai ser submetido. (...) Eu sempre vou ficar focado na análise de requisitos não funcionais. Então eu trabalho bastante no pré-venda aqui né. Então sempre que há uma nova demanda aqui, eu sou envolvido, aí eu trabalho em conjunto com os analistas de requisitos pra tentar estruturar uma visão geral do sistema né, bem nessa fase inicial, e tentar achar possíveis pontos de atenção no sistema. Ah, o sistema é distribuído, vai precisar ter um tempo de vida longo ou curto, quantos usuários, quantos sites vão estar envolvidos, que tipos de tecnologias candidatas a gente pode usar. (...) Então na verdade, a partir da visão geral eu tento encontrar certos pontos de atenção, principalmente focado em requisitos não funcionais. Pra tentar abordar e montar uma estratégia de como se lidar com esses tipos de pontos aí. Por exemplo, os mais comuns são performance e escalabilidade. Mas também pode ter manutenção, certo. Uma série de coisas.****ò*** ï Informante 17, Empresa D.

Segundo o Informante 17, na Empresa D ele tentou formar arquitetos que agregassem características de vários tipos de arquitetos que se vê no mercado, sendo que ele citou arquiteto corporativo, arquiteto de sistemas, arquiteto de soluções e arquiteto de software. Assim, o arquiteto de software desta organização agrega várias atividades de vários dos arquitetos sugeridos pela IASA (e consequentemente dos tipos de arquitetos representados na Figura 4.1), abordando uma gama de aspectos mais ampla do que aquela que normalmente costuma-se estudar na academia. Para o Informante 17, essas divisões de tipos de arquiteto representam uma especialização do papel do arquiteto e são mais úteis para empresas de grande porte. Para ele, a configuração que a Empresa D adota é mais adequada para a sua realidade.

Embora o arquiteto da Empresa D (similar ao arquiteto de soluções) agregue diversas atividades de vários dos papéis de arquiteto da IASA, ainda assim foi possível identificar as atividades do arquiteto de software da Figura 4.1 em um outro ator. Após a criação da visão geral da arquitetura, esta será refinada. Os componentes mais críticos são refinados pelo próprio arquiteto de software (similar ao arquiteto de soluções), porém os componentes intermediários são delegados a desenvolvedores sênior. Assim, nesses casos, esses

desenvolvedores seniores exercem atividades similares às atividades do arquiteto de software da Figura 4.1. O trecho de entrevista a seguir ilustra este fato.

> ***ñ****Na verdade essa arquitetura vai ser refinada a partir do momento da priorização dos maiores riscos. (...) É como se a gente passasse ao lado, certo, e depois fosse detalhando. Detalhando pontos específicos da arquitetura, pontos de persistência, distribuição, mensageria, e assim por diante.(...) Não necessariamente* [esse detalhamento é feito pelo arquiteto]. *Nos módulos que a gente enquadra no nível de dificuldade maior, aí sou eu* [arquiteto] *que desenvolvo. Mas se forem módulos intermediários então aí a gente delega para desenvolvedores sênior.(...) Eles vão fazer* [esse refinamento] *sempre baseando em prova de conceito e depois aprovação da prova de conceito e integração na arquitetura. Esse trabalho de organização entre os arquitetos e os desenvolvedores sênior é bem dinâmico****ò*** ï Informante 17, Empresa D.

4.2.1.2.3. Empresa I

Na Empresa I existe outro caso especial. O projeto no qual o informante desta empresa trabalha começou em 2006, quando era outro arquiteto que trabalhava nele. Este arquiteto foi quem definiu o *design* da arquitetura. O Informante 21 entrou na empresa e no projeto no ano seguinte, quando a arquitetura já estava definida. Ele chegou a trabalhar com o primeiro arquiteto, mas hoje é o único arquiteto neste projeto. Com isso, considerando o projeto como um todo, o arquiteto inicial efetuou as tarefas de arquiteto de soluções da Figura 4.1 e o Informante 21 realiza as tarefas de arquiteto de software da mesma figura. Por outro lado, o trabalho atual do projeto descrito pelo Informante 21 consiste em adicionar funcionalidades ao *framework* ou requisições de mudanças conforme requisições do cliente. Nesse caso, algumas mudanças ou novas funcionalidades podem afetar a arquitetura. Então cria-se um projeto dentro do projeto maior para efetuar essa modificação. Em resumo, o Informante 21 passa a atuar como correlato ao arquiteto de soluções e o ator chamado de líder técnico assume as atividades de arquiteto de software.

Exemplificando o que foi descrito no parágrafo anterior, quando um cliente deseja uma nova funcionalidade ou uma modificação em alguma existente ele mesmo a cadastra na ferramenta que esta empresa usa (Ferramenta I). Após isso, o *Product Onwer* analisa essa requisição e a prioriza de acordo com a necessidade ou não de desenvolver funcionalidades

novas. Caso seja um requisito novo, este passa por analistas de negócio e de sistemas (que podem ser representados pela mesma pessoa) que o analisam em conjunto com o arquiteto. Este vai interagir com vários *stakeholders* para adquirir as informações necessárias para definir a visão geral da arquitetura, como ilustrado no seguinte trecho de entrevista:

> **ñ***Aí* [o arquiteto correlato ao arquiteto de soluções] *conversa com o product owner, com o analista pra entender bem a solução, com os desenvolvedores e até mesmo com os testadores né?***ò** ï Informante 21, Empresa I.

Com essa análise feita pelos analistas e, principalmente, pelo arquiteto, quando um requisito provoca uma mudança muito grande na arquitetura, é criada uma comissão de duas pessoas, encabeçada pelo arquiteto (correlato ao arquiteto de soluções), para informar a gerência sobre o fato, como o Informante 21 informa a seguir.

> **ñ***Então, dentro do projeto a gente tem uma espécie de comissão, né. Eu e outra pessoa que decidimos quando uma operação é bem drástica e a gente normalmente leva ao conhecimento da gerência***ò** ï Informante 21, Empresa I.

Dessa forma, o arquiteto de software (similar ao de soluções) realiza essa modificação na arquitetura e a passa para o líder técnico (que por sua vez é similar ao arquiteto de software da Figura 4.1). Este é um papel que pode ser desempenhado por um desenvolvedor que tenha uma boa visão do produto todo e assim vai servir como um referencial para o time de desenvolvimento, especialmente quando o arquiteto não pode ajudá-los mais de perto. O próximo trecho de entrevista descreve o papel de líder técnico.

> **ñ***Ele* [o líder técnico] *é um desenvolvedor, mas ele também tem bastante experiência com arquitetura e ajuda a tirar dúvidas de outros desenvolvedores, as vezes até pra resolver um problema que eu não estava conseguindo visualizar a solução. Ele vai lá e pode colaborar nisso. (...) Ele colabora quando o arquiteto não tem disponibilidade pra ajudar. Serve como referência.***ò** ï Informante 21, Empresa I.

É importante citar que, em alguns casos pode ser que o próprio arquiteto desempenhe esse papel de líder técnico e em outros quem faz isso é um desenvolvedor que possui um grande conhecimento e uma visão geral do produto que eles estão produzindo. Assim, nas duas possibilidades de interpretação da configuração desta empresa, ela apresenta sempre uma divisão nas atividades de arquitetura de modo que existe um ator sempre focando na solução geral e outro com foco no refinamento da mesma.

4.2.1.3. Tipos de arquitetos não definidos

Por fim, o último grupo compreende as Empresas F e G. Estas empresas apresentam algumas similaridades. Ambas se chamam de fábricas de software e possuem apenas um cliente: a Empresa F é a divisão de TI de uma organização para soluções para cartões de crédito e trabalha apenas para ela, enquanto a Empresa G é uma filial regional de uma empresa estatal criada especificamente para oferecer apoio a um grande cliente local. Ambas também trabalham em um único projeto por vez. O time de arquitetura delas é composto por quatro arquitetos que são simplesmente chamados de arquitetos de software: eles não possuem atividades de arquitetura diferenciadas formalmente pela empresa.

Neste grupo não foi possível identificar atividades de um arquiteto *enterprise* definidas dentro das empresas. Já as atividades dos outros dois tipos de arquiteto são bem identificadas, mas elas são intercambiáveis entre os arquitetos de software da empresa. Em outras palavras, elas não são realizadas por um tipo de arquiteto definido em cada projeto e nem mesmo o arquiteto que as realiza é designado previamente para isso. O arquiteto (ou às vezes mais de um) do time de arquitetura que melhor se identifica com as atividades de arquiteto de soluções do projeto, especialmente as relacionadas com criar a visão geral da arquitetura, toma para si este trabalho. Os outros automaticamente passam a realizar atividades de arquiteto de software (da Figura 4.1), refinando a visão geral proposta pelo primeiro.

Outra característica comum neste grupo é que os arquitetos (tanto de solução quanto de software) têm pouco ou nenhum contato com clientes. Na Empresa F essa interação não é possível devido a um regulamento interno, enquanto que na Empresa G ela pode até acontecer, mas não é comum. Em ambas as empresas, quem interage com o cliente são os analistas que desenvolvem as especificações de requisitos e as repassam para os arquitetos.

Em resumo, em cada projeto há sempre um arquiteto que trabalha e cria a solução geral da arquitetura, trabalhando como um arquiteto de soluções (embora não de maneira formal), enquanto os outros arquitetos assumem as atividades típicas de um arquiteto de software, trabalhando próximos aos desenvolvedores e desenvolvendo componentes críticos. Além disso, o fluxo de informações vindas do cliente sempre passa antes pelos analistas e os arquitetos não tem contato com o cliente. Os arquitetos também servem como uma ponte entre o time de desenvolvimento e o time de análise em ambas as empresas, mesmo considerando as diferenças já descritas. A interação com gerentes também é realizada principalmente pelos arquitetos, mas as empresas também dispõem de reuniões nas quais outros papéis também interagem com a gerência.

A seguir as instanciações dos tipos de arquitetos das duas empresas serão apresentadas para exemplificar o que foi descrito nos parágrafos anteriores.

4.2.1.3.1. Empresa F

Como resumido na seção anterior, os arquitetos desta empresa dividem as atividades de arquitetura entre si de forma espontânea por uma questão de identificação com a solução do projeto, como exemplificado no trecho de entrevista a seguir:

> ***ñ****Tem um deles que vê o todo. (...) Então, a gente pega esse todo e vai refinando. Isso realmente acontece, mas não quer dizer que esse* [mesmo arquiteto] *seja obrigado a ver o todo. Pode ser que num determinado projeto eu veja o* [design] *geral* [da arquitetura] *e a gente monte essa arquitetura.****ò ï*** *Informante 9, Empresa F.*

Assim, sempre há um arquiteto responsável pela solução geral e outros responsáveis pelo refinamento dela. As atividades de arquiteto *enterprise* não foram encontradas nesta empresa: não existe um papel com atividades de arquitetura que participe das reuniões iniciais de definição do contrato nem das reuniões de definição do escopo do produto devido a um regulamento interno. Os arquitetos de software da empresa F nem ao menos interagem com o cliente; esta interação é feita através dos analistas que enviam as especificações prontas para os arquitetos. Os seguintes trechos de entrevista ilustram estes fatos.

ñNós não lidamos com o cliente. Os nossos clientes, dentro da Empresa G são os analistasò ï Informante 9, Empresa F.

ñNós não estamos lá na parte quando está começando o projeto. A gente já pega o projeto com ele mais ou menos definido. Está mais ou menos especificado. A gente só pode ir em alguma reunião lá... não são em todos os projetos que nós temos reuniões com os analistas. Então, eles só nos enviam a tarefa e a gente desenvolve em cima disso. Mas, quando o analista já vê, já verifica que vai ser algo crítico, ele chama um arquiteto pra ver se a solução que ele está propondo é viável ou é a melhor. Então, essa comunicação existe esporadicamenteò ï Informante 9, Empresa F.

Apesar de os analistas serem o contato dos arquitetos da empresa (similares tanto ao de soluções quanto ao de software) com o cliente, a interação entre os dois grupos é esporádica. Ela acontece quando os analistas verificam que um ponto da especificação será crítico e chamam um arquiteto para averiguar se a solução proposta é viável. Além disso, os arquitetos da empresa também fazem a ponte entre analistas e desenvolvedores. Por exemplo, quando um desenvolvedor tem alguma dúvida sobre uma funcionalidade ou requisito ele contata um dos arquitetos da empresa que, se por algum motivo não puder resolver sozinho, entra em contato com os analistas. Ambas as interações não são frequentes, como exemplificado no seguinte trecho de entrevista:

ñQuando um desenvolvedor tem uma dúvida, normalmente antes de falar com o analista ele vem até o arquiteto: ó você inspecionou isso, como você pensou que essa parte deveria ser desenvolvida?â Então se você não inspecionou bem ou alguma coisa do gênero, você leva para o analista porque é alguma coisa relacionada a um requisito funcional que você deixou passar: óestava claro pra mim, mas agora passando pra código não está tão claroâ Então você faz contato, você abre um canal com o analista de sistema, uma vez que você não conseguiu resolver sozinho.ò ï Informante 10, Empresa F

4.2.1.3.2. Empresa G

Quando a Empresa G começou o desenvolvimento do seu projeto atual (que foi o que motivou a criação desta filial da empresa), havia apenas um arquiteto (Informante 12). Logo

depois um segundo (Informante 11) foi contratado e eles desenvolveram a arquitetura em conjunto. Os outros dois arquitetos entraram na empresa mais tarde (o Informante 13 foi o último). Assim, neste início de projeto, os Informantes 11 e 12 exerceram as atividades de arquiteto de soluções da Figura 4.1 para o projeto todo, definindo a solução arquitetural do projeto todo.

Hoje, um dos arquitetos iniciais (Informante 12) se especializou nas integrações que esse sistema tem com outros sistemas do cliente e trabalha somente com isso. Por outro lado, o Informante 11 continua exercendo atividades do arquiteto de soluções, não se especializando em um único módulo do sistema; ele trabalha e fornece apoio para todos os módulos. Enquanto isso, os outros dois arquitetos de software da empresa realizam atividades do arquiteto de software da Figura 4.1.

O projeto que a Empresa G desenvolve é dividido em diversos módulos. A arquitetura do projeto como um todo foi criada pelos Informantes 11 e 12. Após isso, cada módulo é como se fosse um projeto novo com sua própria arquitetura, requisitos e períodos de desenvolvimento. No momento da definição desta visão geral, o Informante 13 participou apenas da definição da tecnologia, não participando da modelagem da arquitetura. Hoje ele realiza atividades de arquiteto de software (ver Figura 4.1) em um dos módulos do projeto. O trabalho atual dele consiste em analisar os requisitos e verificar como eles se encaixam na solução de arquitetura proposta, ou seja, instanciar os requisitos na arquitetura de forma a refiná-la antes de os desenvolvedores a detalharem.

> ***ñ****Meu trabalho hoje é mais encontrar soluções. Casar tanto a solução de negócio com a solução que a gente tem hoje desenvolvida pra este módulo. Entender o que é a regra de negócio que é o que o cliente precisa, o que ele precisa do Módulo G, o que ele precisa pra atender a demanda da Atividade G e utilizar isso no nosso módulo que está sendo desenvolvido. Esse trabalho em cima de mais ou menos isso. Entender um lado a nível de negócio e casar isso com a nossa solução****ò** ï Informante 13, Empresa G.

> ***ñ****É como se fosse instanciar as necessidades de negócios do cliente na solução que a gente tem. E visualizar de repente algumas mudanças na arquitetura que está proposta****ò*** ï Informante 13, Empresa G.

A interação com o cliente nesta empresa geralmente também acontece através dos analistas, apesar de neste caso os arquitetos ainda manterem um contato pequeno com o cliente quando necessário; o mais comum é que ela seja feita por intermédio dos analistas.

4.3. Interdependências dos arquitetos entre si e com outros *stakeholders*

Nas seções anteriores a forma como as empresas organizam seus tipos de arquitetos foi descrita e, com isso, o fluxo de informações foi implicitamente discutido, uma vez que as principais interações desses tipos de arquitetos também foram identificadas. Através dessas interações os arquitetos (qualquer que seja o tipo) recebem diferentes informações que são úteis para a definição da arquitetura, seja no nível organizacional ou no nível técnico. Após trabalhar com essas informações, eles precisam repassá-las para vários outros *stakeholders*, ou porque estes precisam delas para realizar seu trabalho ou para devolver *feedback* sobre o projeto, de forma que com isso eles ajudam a criar o conhecimento comum que é tão importante em projetos de desenvolvimento de sistemas. Dessa forma, os arquitetos dependem das informações que recebem e os outros *stakeholders* dependem das informações que os arquitetos repassam, configurando assim um cenário de interdependência entre eles. Considerando os vários tipos de arquitetos presentes nas empresas, existem ainda as interdependências entre os próprios arquitetos, uma vez que a arquitetura final no sistema vai ser formada a partir de informações do trabalho de todos os tipos de arquiteto. Com isso, as seções seguintes apresentam os aspectos mais importantes dessas interdependências que foram encontrados durante a pesquisa nas empresas.

4.3.1. Tipos de arquitetos e a influência de outros *stakeholders* e do contexto

A Figura 4.1 ilustrou a configuração genérica dos tipos de arquitetos que se encontram nas empresas e as seções 4.2.1. e subseções descreveram a classificação das empresas e as configurações específicas de cada uma delas. Além disso, as principais interações de cada tipo de arquiteto também foram apresentadas. Como será ilustrado nesta seção, a divisão em diferentes tipos de arquitetos representa uma forma de dividir (i) as atividades relacionadas à arquitetura de TI e (ii) a comunicação relacionada à ela, uma vez que cada arquiteto interage principalmente com grupos específicos de *stakeholders*. O Informante 1 da Empresa A resume a divisão das interações quando a empresa possui os 3 diferentes tipos de arquitetos, caracterizando assim o fluxo representado na Figura 4.1:

ñ*Só o que diferencia é o* ó*nível*ô *dos stakeholders. O enterprise architect* [arquiteto enterprise] *fala mais com CIOs e CEOs, mais nesse nível. Já o arquiteto de soluções de repente vai estar falando com um arquiteto do outro lado* [empresa cliente], *ou então com um diretor do outro lado. E o arquiteto técnico* [fala] *mais com os desenvolvedores. São todos os níveis, só que o nível é diferente em cada um dos pilares* [tipos de arquitetos]. *(...) E eles* [os arquitetos] *se comunicam entre si*ò ï Informante 1, Empresa A.

Como é possível perceber, os arquitetos são responsáveis pela interação com diversos grupos de *stakeholders* e vão usar as informações obtidas deles no seu trabalho e também repassar informações para eles. Isso acontece desde o início do projeto e permeia todo o processo de desenvolvimento do software. Para explicar a influência dessas interações e do contexto no trabalho dos arquitetos, serão discutidos a seguir trechos de entrevistas onde os informantes as mencionam.

No início dos projetos, quando as empresas possuem um arquiteto *enterprise*, há a interação ao menos do arquiteto *enterprise* com gerentes e clientes para definir os termos do contrato. Essa interação é interessante pois o arquiteto traz informações importantes sobre a viabilidade da solução, os padrões da empresa e a experiência técnica que pode ser usada em estimativas, por exemplo. E, ao mesmo tempo em que ele fornece essas informações mais técnicas, ele também adquire as informações organizacionais que são importantes para a arquitetura, como por exemplo algum legado que o cliente possua que precisará ser integrado com a solução que a empresa desenvolverá. O trecho de entrevista a seguir exemplifica essa interação entre arquiteto e gerente, especialmente no início do projeto.

ñ*Assim, ele* [o arquiteto enterprise], *junto com o gerente da unidade de negócio mais com o cara responsável de marketing, define qual seria a confecção do produto, a ideia do produto. A partir dali, daquela definição, daquela ideia é que saem as iniciativas de trabalho. (...) Então é assim, ele* [o arquiteto enterprise] *define, mas assim, ele não é o único responsável, porque na última instância o responsável por aquilo é o gerente da unidade de negócio, porque ele quem tem o dinheiro e, normalmente, esse cara é chefe desse arquiteto, então não passa direto*ò ï Informante 1, Empresa A.

A Empresa A possui o tipo de arquiteto *enterprise* bem definido e é este tipo de arquiteto que geralmente participa dessa etapa inicial. Entretanto, na Empresa C, por exemplo, o tipo de arquiteto correlato ao arquiteto de soluções é quem desempenha algumas das atividades do arquiteto *enterprise*. O trecho de entrevista a seguir ilustra como, mesmo sem a presença de um arquiteto *enterprise* bem definido, estas atividades de arquitetura iniciais são realizadas na Empresa C (no caso através do papel correspondente ao arquiteto de soluções dessa empresa). Nele é possível notar que informações do plano do projeto, marcos e datas são importantes para a criação da visão geral da arquitetura.

> ***ñ****Talvez* [o arquiteto equivalente ao de soluções da IASA] *seja a única figura que vai do início ao fim do projeto, até a parte de correção de bugs e tudo mais lá no fim. Então, o que acontece? Ele acaba tendo que pegar o conhecimento do plano de projeto, que é um gestor que vai estar trabalhando, que é um gerente que vai estar trabalhando, te apoiando, vai pegar marcos, datas, criticidades, vai ter o cliente lá do outro lado dizendo o que ele acha que é importante pro sistema ter****ò** ï Informante 5, Empresa C.

Nesta etapa inicial de definição dos produtos que serão desenvolvidos existe um outro grupo que geralmente está presente, especialmente em empresas maiores. Este grupo é chamado de área de *marketing* ou área de produto da empresa. Ele é responsável por capturar as tendências de mercado, analisar a concorrência e criar novas necessidades de mercado onde a empresa possa atuar. Dessa forma, as informações que esse grupo de *stakeholders* apresenta também influenciam na arquitetura (de TI e de software). Eles também interagem com os arquitetos, pois estes precisam saber das tendências do mercado e dos produtos da concorrência para que o projeto produza um produto que possa ser competitivo ou então explorar uma nova área do mercado. O Informante 1 da Empresa A descreve o trabalho do time de *marketing* da sua empresa e o Informante 7 da Empresa E descreve o trabalho da área de produto da sua empresa nos exemplos a seguir:

> ***ñ****O marketing traz os requisitos dele mesmo. Ele é uma entidade própria (...) tem uma lista de requisitos que quer atender, porque o cara de marketing* [que interage com o arquiteto enterprise] *está bem atento ao que está acontecendo no mercado, principalmente o que está ocorrendo nas outras empresas, então ele consegue apontar e dizer* ***ñ****olha, eu*

*não estou conseguindo vender esse produto porque o competidor lá tem essa funcionalidade e o cara está me anulando dessa forma, então eu preciso fazer alguma coisa pra acabar com essa vantagem que o cara tem***ò**. *Então é isso que o marketing traz aqui pra gente***ò** ï Informante 1, Empresa A.

ñ*A área de produtos trouxe em linhas gerais o que ela queria (...). A gente pegou isso e aí detalhamos os requisitos e validamos todos juntos* [equipe multidisciplinar]**ò** ï Informante 7, Empresa E.

Geralmente, a interação com este grupo de *stakeholders* acontece nas etapas iniciais dos projetos, durante a definição do produto, quando ele acrescenta aos requisitos da organização (relativos a padrões, por exemplo) e do cliente (relativo às funcionalidades do produto) os requisitos de mercado que ele coleta. Entretanto, este grupo influencia o trabalho de arquitetura de todos os tipos de arquiteto, pois seus requisitos precisam entrar na definição geral da arquitetura e precisam ser refinados para serem desenvolvidos. O Informante 1 é um arquiteto técnico na Empresa A, equivalente ao arquiteto de software da Figura 4.1, e descreve brevemente no trecho de entrevista a seguir essa influência ao responder como chegam as informações que serão desenvolvidas pelo seu time no Brasil:

ñ*Vem a carta de* **ó***papai noel***ô** *do marketing dizendo assim* **ó***ah eu gostaria que fizesse isso, isso e isso***ô** *Aí a gente vai pegar aquilo ali e começar a transformar, desenvolvendo use cases pra isso. Trabalha os use cases e aí vai tirar os requisitos. Aí os requisitos são desenvolvidos***ò** ï Informante 1, Empresa A.

Outra interação muito comum dos arquitetos acontece com os clientes. Apesar desta interação não existir em empresas como a Empresa F, ela é muito importante, uma vez que o produto será desenvolvido para o cliente (que pode ser interno, como a área de produto da empresa), logo ele deve atender as necessidades deste cliente e a arquitetura é fundamental para isso. Assim, também é preciso que os arquitetos interajam com os clientes para capturar informações, especialmente sobre os requisitos não funcionais, que serão fundamentais para a arquitetura e também precisam contribuir para a compreensão do cliente sobre o que vai ser produzido no projeto. Os tipos de arquiteto *enterprise* e de solução são os que mais interagem

com estes *stakeholders*. A interação com o *enterprise* ocorre na definição do produto que será desenvolvido, como descrito anteriormente, já a interação com o arquiteto de soluções é mais frequente, pois este precisa de requisitos, especialmente os não funcionais, que só o cliente pode prover (os quais são capturados em conjunto com os analistas). Dessa forma, em algumas empresas como a Empresa A, este tipo de arquiteto encontra-se inclusive dentro do cliente, como o trecho de entrevista a seguir ilustra, capturando essas informações que precisará para o seu trabalho e também servindo como fonte de informações para outros *stakeholders* devido a essa interação.

> ***ñ****O arquiteto de solução está muito próximo do cliente. Então assim, normalmente um arquiteto de solução é um cara que vive no desenvolvimento dentro do cliente, ele não vive na Empresa A. Então assim, a Empresa A tem uma conta com um cliente grande, então esse cara, esse solution architect, é o cara que representa a solução da Empresa A dentro do cliente. Então assim, não é que o cara saiba mais, mas como ele vive junto do cliente ele consegue colher feedback e entender bem a maneira que o cliente usa aquele produto. É por isso que ele é uma fonte importante de informação. Porque ele consegue nos alimentar de maneira muito eficiente pelo posicionamento dele****ò ï*** *Informante 1, Empresa A.*

Uma interação tão importante quanto com os clientes é a que acontece com os desenvolvedores. Se por um lado os clientes são quem fornece os requisitos dos sistemas, por outro os desenvolvedores são quem os desenvolve. E os arquitetos trabalham entre esses dois extremos, logo precisam interagir com ambos. Com a divisão de tipos de arquitetos, essa interação é separada, pois o arquiteto de soluções (e o *enterprise* no início do projeto) geralmente interage mais com os clientes e o arquiteto de software com os desenvolvedores. Além disso, diferente da interação com cliente que não é comum nas empresas do grupo ñtipos de arquiteto não definidosò, a interação com os desenvolvedores é presente e forte em todas as empresas pesquisadas. Os trechos de entrevista a seguir ilustram essa interação:

> ***ñ****A gente tem feito reuniões locais aqui com o time de desenvolvimento pra que o time de desenvolvimento tenha a oportunidade de fazer perguntas, caso eles não tenham entendido bem o que está escrito; o que se mostrou bem importante, porque normalmente tem pontos*

ali que são levantados por eles que muitas vezes eu como arquiteto [similar ao de software] *não estou a par, que são detalhes de implementação, detalhes de interfaces de componentes que eles, como são muito mais próximos disso, conseguem contribuir de forma muito mais precisa do que eu consigo***ò** ï Informante 1, Empresa A.

ñ*A rotatividade do desenvolvedor é muito grande. Então, a gente tem que estar sempre com eles, tem que estar sempre buscando saber se ele está com problemas, saber se o que ele está fazendo já não existe, pra não ter que ficar replicando o código. Bom, então, a nossa gerência nos cobra bastante isso - que a gente fique tendo um contato direto com os nossos desenvolvedores pra diminuir bastante o retorno de tarefas, ou correções de tarefas. Isso, lá dentro, no meu caso, eu acho que eu sou dos que mais conversa com eles porque, a minha parte dentro da arquitetura já não trabalho muito, trabalho mais a parte de coaching dos desenvolvedores***ò** ï Informante 9, Empresa F.

O trecho de entrevista a seguir é de um informante da Empresa E, onde muitas das atividades são realizadas por equipes multidisciplinares. Nele é possível perceber que a interação com os desenvolvedores é tão importante que eles participam nesta empresa inclusive da definição da visão geral da arquitetura, juntamente com o arquiteto (equivalente ao de soluções) e a equipe de *delivery* de serviços.

ñ*Então a gente pega esses 3 perfis* **ï** *service delivery, a equipe de desenvolvimento* [que inclui o líder técnico, equivalente ao arquiteto de software nessa empresa] *e o arquiteto* [de soluções] **ï** *pega os requisitos, coloca no quadro e começa a desenhar as caixinhas, como elas se relacionam, identifica quais os componentes vão ser reutilizados, quais vão ser construídos... E dos que vão ser construídos a gente detalha dentro dele como é que vai funcionar. (...) Quando a gente tem a lista de requisitos, o arquiteto* [de soluções] *é uma peça, assim como o time de desenvolvimento e o service delivery, ele não está acima nem entre, ele está junto. Papéis complementares. Eu ainda defendo muito essa ideia de complemento e não serialização***ò** ï Informante 7, Empresa E.

Resumindo: os arquitetos (considerando todos os tipos) interagem com diferentes grupos de *stakeholders* do início ao fim do projeto, e todos esses grupos fornecem

informações importantes para a realização do trabalho de arquitetura, assim como os arquitetos retornam para eles as informações adequadas, ajudando com isso a criar um conhecimento comum sobre o projeto. O trecho de entrevista a seguir destaca este fato:

> ***ñ****Talvez* [o arquiteto equivalente ao de soluções] *seja a única figura que vai do início ao fim do projeto, até a parte de correção de bugs e tudo mais lá no fim. Então o que acontece? Ele acaba tendo que pegar o conhecimento do plano de projeto, que é um gestor que vai estar trabalhando, que é um gerente que vai estar trabalhando, te apoiando, vai pegar marcos, datas, criticidades, vai ter o cliente lá do outro lado dizendo o que ele acha que é importante pro sistema ter, depois o analista de negócios também te ajudando, apoiando a detalhar as regras. O que acaba acontecendo? Ele acaba tendo que criar uma linguagem que o cliente consiga daí entrar* [fornecer], *consiga ser traduzido isso pra dentro do sistema ou dentro da arquitetura que a gente está instanciando, ou pelo menos, criar critérios pra hora de instanciar a arquitetura pra que essas coisas aconteçam****ò*** ï Informante 5, Empresa C.

Dessa forma, como é possível notar, essas interações permeiam todo o projeto, desde a definição do contrato até o desenvolvimento e entrega do produto final. Com isso, é necessário que os arquitetos participem do projeto todo, algo já sugerido na literatura (Hofstader, 2008)(Taylor *et al.*, 2010). O trecho de entrevista a seguir ilustra o porquê dessa necessidade de os arquitetos estarem presentes ao longo de todo o projeto sob o ponto de vista das interações com diferentes *stakeholders* e da difusão de informações relacionadas a elas.

> ***ñ****O arquiteto de software tem que estar presente em todas as fases do projeto. Não pode ser: pega requisito, monta um documento e passa pro desenvolvimento e vai pra outro projeto. Porque quem vai desenvolver o projeto não é ele, quem vai desenvolver o projeto não tem o mesmo conhecimento que ele. Não quer dizer que tenha menos conhecimento, mas tem um outro conhecimento e pode ser que ele entenda uma coisa de outra maneira. No final, muito provavelmente os requisitos vão ter sido atendidos, mas talvez não da maneira como ele tinha pensado. E pode ser que ele pensou daquela maneira pra que isso que vai ser construído seja usado em um outro projeto que ele está trabalhando, então quando ele receber aquele produto não vai servir****ò*** ï Informante 7, Empresa E.

Quando existem os diferentes tipos de arquitetos, eles tendem a dividir essas interações com os diversos *stakeholders* e, com isso, um arquiteto (*enterprise*) foca no nível gerencial, outro (de software) foca mais no nível técnico e o terceiro (de soluções) tem o foco em um nível intermediário aos dois primeiros. Assim, mesmo que algum deles não atue no projeto inteiro, ainda é possível realizar uma adequada difusão de informações e manter um conhecimento comum sobre o projeto, uma vez que sempre vai haver pelo menos um tipo de arquiteto envolvido e que este interage com os outros tipos de arquiteto (o que será descrito na seção seguinte).

A presença dos arquitetos durante o projeto todo é importante também devido a influência que o contexto da empresa exerce no trabalho deles e na própria arquitetura. No trecho de entrevista a seguir, essa influência do contexto é exemplificada. Nele o informante fala sobre investimentos e relações comerciais, informações que recebe no início do projeto, quando a empresa fecha o negócio. É importante lembrar que a Empresa C não possui o tipo de arquiteto *enterprise* definido, mas já apresenta atividades características dele sendo desenvolvidas pelo arquiteto similar ao arquiteto de soluções, como é possível perceber neste trecho de entrevista.

> ***ñ****O arquiteto* [de soluções] *tem uma função bem à parte, até bem separada disso tudo: ele pega a ideia geral do projeto, o que vai ser o projeto. Ele entende qual é o nível de investimento que vai ser feito no projeto, e isso às vezes não depende do analista em si, depende da relação comercial que é estabelecida. A gente já recebe algumas informações lá na hora que está fazendo o contrato (...). Ele* [o arquiteto de soluções] *não se prende ainda no desenvolvimento em si. Ele define como as peças vão se ligar, qual é o produto mais adequado pra cada situação****ò*** ï Informante 4, Empresa C.

Além da presença de um arquiteto na definição do produto ser importante devido ao contexto, é importante também que essas informações iniciais sejam repassadas aos outros arquitetos, pois o contexto da empresa também pode influenciar o trabalho dos arquitetos mais ligados ao desenvolvimento de um novo produto. No primeiro trecho de entrevista abaixo se observa que o arquiteto de sistemas (também chamado de responsável técnico e

equivalente ao arquiteto de soluções) é responsável por pesquisar os produtos da Empresa E e os produtos que ela usa (incluindo software livre) para assim reutilizá-los nos novos projetos. Esse aspecto já foi retratado no último trecho de entrevista do Informante 7 desta mesma empresa, do qual a parte relativa a este assunto foi repetida como o segundo trecho de entrevista a seguir. Dessa forma, o *background* da empresa influencia no trabalho também dos arquitetos de soluções.

> ***ñ****Ele tem um tempo maior de estudo para novas tecnologias, tem um tempo maior para conhecer a estrutura da Empresa E, conhecer outros produtos. (...) Ele não está lá pra definir que interação com a área de produtos externos, ele está lá para realmente mastigar o que o projeto tem pra ser feito:* ***ó****vamos usar esse componente****ô****,* ***ó****já tem um produto de software livre que faz essa funcionalidade, a gente só tem que adaptar ou estender****ô****, etc****ò*** ï Informante 8, Empresa E.

> ***ñ****Quem vai desenvolver o projeto não é ele* [arquiteto de uma forma geral]*, quem vai desenvolver o projeto não tem o mesmo conhecimento que ele. (...) E pode ser que ele pensou daquela maneira pra que isso que vai ser construído seja usado em um outro projeto que ele está trabalhando, então quando ele receber aquele produto não vai servir* [caso o arquiteto não acompanhe o projeto todo]***ò*** ï Informante 7, Empresa E.

4.3.2. Interdependências entre os tipos de arquitetos.

A sessão anterior tratou da influência do contexto da organização na atividade dos arquitetos e da interação deles com os diferentes *stakeholders* . Quando existem os três tipos diferentes de arquitetos discutidos aqui essa comunicação e interação tende a ser dividida entre eles, como o Informante 1 afirmou e foi representado na Figura 4.1: o arquiteto mais organizacional (*enterprise*) interage mais com os *stakeholders* mais gerenciais, como gestores, CIOs e CEOs; o arquiteto mais técnico (arquiteto de software) interage mais com os desenvolvedores; e o arquiteto ñintermediárioò (de soluções) fica mais próximo do cliente, estando inclusive muitas vezes localizado dentro da empresa cliente, como na Empresa A. Assim, como estes arquitetos recebem tipos de informações diferentes é necessário que eles interajam entre si para trocar essas informações e assim repassá-las para os outros *stakeholders*. Dessa forma,

em adição à interação que os arquitetos têm com diferentes *stakeholders*, é importante considerar que eles também interagem entre si. *Em outras palavras, o trabalho de cada um deles não é independente um do outro, suas atividades estão interconectadas; eles dependem uns dos outros para executar suas diferentes atividades.* Eles recebem informações de vários grupos de *stakeholders*, realizam o seu trabalho com elas usando informações adicionais que eles possuem, e repassam um tipo diferente de informação, modificado pelo seu trabalho, para o próximo arquiteto. O exemplo a seguir descreve como, também devido a essas interações com outros *stakeholders*, a interação entre os diferentes tipos de arquitetos é importante.

O arquiteto *enterprise* desenvolve os princípios arquiteturais que serão usados na organização como um todo (isto é, em todos os projetos da mesma) e trabalha também nas definições dos contratos com clientes. Para criar estes princípios, ele interage com *stakeholders* da alta gerência e para fechar os contratos, ele também interage com os clientes. Assim, o arquiteto *enterprise* adquire, a partir dessas interações com esses *stakeholders*, informações que são importantes para realizar as suas próprias tarefas. O resultado dessas atividades é muito importante para os arquitetos de soluções, pois eles precisam estar cientes dos princípios arquiteturais da organização e dos termos do contrato do projeto para produzir a arquitetura de cada projeto no qual eles trabalham. Em outras palavras, o arquiteto de soluções precisa considerar e obedecer aos princípios arquiteturais e os termos do contrato que foram definidos pelo arquiteto *enterprise* para produzir uma arquitetura que esteja em conformidade com estes dois aspectos, como exemplificado no seguinte trecho:

> ***ñ****Quando começa a subir o nível, esse cara começa fica mais próximo à parte de marketing e a parte de solução é mais complexa. O cara* [arquiteto enterprise] *até fala de requisito, mas fala de requisitos bem aberto assim, não é nada definido, são requisitos do tipo* ***ó****ah, eu acho que a gente precisa aumentar a escalabilidade do produto* ***ô*** *e não diz o que da escalabilidade a gente vai aumentar e também, tipo, como vai ser feito isso, porque isso vai descendo* [para os outros tipos de arquiteto]. *Mas assim, ele elabora* [os requisitos] *junto com os clientes, (...) e aí ele diz assim* ***ó****é importante que a gente tenha uma escalabilidade maior* ***ô*** *e aí o pessoal que está mais embaixo* [arquiteto de soluções e de software] *consegue entender o que a gente vai crescer* [arquiteto de soluções] *e como que a gente vai crescer, que é a parte do technical architect* [correlato ao arquiteto de software da Figura 4.1] *que eu te falei que é onde eu atuo. Então assim, essa é a diferença, o*

enterprise architect, que é o nível mais alto, vai estar olhando basicamente como é que está a indústria de forma geral, em cima disso ele vai estar propondo soluções, pesquisando soluções e apontando direções, então ele vai ver tendências de mercado assim, óah, o mundo inteiro hoje fala com VMWare, virtualização é importante, então tem que aumentar o suporte a virtualizaçãoô, aí o cara [arquiteto de soluções] *vai dizer assim órealmente tem uma quantidade de clientes que demanda issoôe a gente vai nessa direção. Aí o cara mais em baixo* [arquiteto de software] *vai focar. Essa é a divisão com relação a escopo: o enterprise achitect é muito mais amplo e muito menos específico, já o technical achitect, que é o cara mais baixo, está mais focado no produto específico e aí ele vai ver como que aquele produto vai atender as direções que o cara de alto nível* [o arquiteto enterprise] *está apontando. Existe uma matriz que é feita com relação ao arquiteto responsável* [arquiteto enterprise] *por isso e o gerente da unidade de negócio que financia pra conseguir ter esse produto. Então assim, tem que ter um alinhamento dessas duas dimensões pra que o produto seja realmente acordado e a partir dali cria um roadmap dos produtos onde esse arquiteto principal* [o arquiteto enterprise] *vai estar focado em garantir que aquilo que foi direcionado está correto, validando com os clientes (...) enquanto os outros* [os arquitetos de soluções e de software] *estão mais focados em realmente produzir o produto pra conseguir cumprir o aquele roadmap que foi acordado.òï Informante 1, Empresa A.*

Enquanto isso, o arquiteto de soluções cria a solução geral da arquitetura e depois essa solução é dividida para ser refinada e desenvolvida. Cada ñparteò dessa solução geral é refinada por um arquiteto de software que trabalha próximo e alinhado com os desenvolvedores (que são os *stakeholders* que desenvolverão estas ñpartesò). Para ser mais específico, os trechos de entrevista a seguir são exemplos de como o arquiteto de soluções influencia o trabalho do arquiteto de software:

ñ*Ele* [arquiteto de soluções] *não é um cara que vai fazer modelagem de sistema, é um cara assim que ele quer saber da onde veio, pra onde vai, onde é que isso se encaixa no mundo, por que meu cliente vai querer usar ou não vai querer usar, quais são os use cases de alto nível. Ele está flutuando lá com os clientes, dizendo onde é que estão as possibilidades e só*

grita pra nós [os arquitetos de software] *aqui ócara, é isso que tem que ser feitoôò* ï Informante 1, Empresa A.

ñ*Ele* [o documento de arquitetura produzido pelo arquiteto de soluções] *serve pra primeiramente, definir ao que o sistema tecnicamente vai suprir. Aí tem os componentes centrais da arquitetura que o projetista* [correlato ao arquiteto de software da Figura 4.1] *que vai desenvolver depois, é quem vai pegar algum requisito funcional, vai olhar pra isso e vai encaixar nessa arquitetura. Ou seja, vai pegar dali alguma coisa que já existe na arquitetura e vai encaixar isso pra que ele possa a partir daí defender o projeto e desenvolver esse requisitoò* ï Informante 5, Empresa C.

Finalmente, como este último trecho de entrevista acima ilustrou, o arquiteto de software precisa da definição da visão geral para realizar o seu trabalho, ou seja, para refinar essa visão no que tange os componentes da sua localidade e guiar os desenvolvedores durante a implementação. Na verdade, não é incomum que um projeto tenha mais de um arquiteto de software; eles podem ser um para cada ñparteò da solução geral dividida e estar espalhados por várias filiais da empresa no mundo. O seguinte trecho de entrevista trata dessa divisão da visão geral da arquitetura em várias frentes onde os arquitetos de software vão trabalhar.

ñ*Por exemplo, óah, a Empresa A fechou contrato com uma grande mineradoraô, tem um cara ali que é o responsável por aquele contrato como arquiteto* [enterprise], *só que esse contrato envolve várias áreas. Então, vai envolver uma parte de servidores, uma parte de armazenamento, uma parte de redes. Então cada uma dessas partes tem esses solutions architect* [arquiteto de soluções], *que trabalham em si focados naquela parte ali, dentro dessa parte. A área de armazenamento por si só é grande, então dentro dessa parte tem os outros technical achitects* [similar ao arquiteto de software da Figura 4.1] *que trabalham dentro dos produtos que vão formar aquele nicho que é representado pela área de armazenamento do projetoò* ï Informante 1, Empresa A.

Assim, como é possível perceber, o trabalho dos diferentes tipos de arquitetos possui várias dependências, assim como os artefatos que eles produzem. Entretanto, é importante ressaltar que os diferentes tipos de arquitetos não trabalham em sequência, como em um

modelo em cascata ou uma hierarquia. Ao invés disso, esses diferentes arquitetos *colaboram* durante o projeto. Essa situação é claramente ilustrada no trecho de entrevista a seguir, no qual o Informante 1 ressalta justamente as interconexões entre os diferentes tipos de arquitetos:

> ***ñ****Uma solução feita geralmente pra um cliente* [externo], *o enterprise vai estar lá em cima envolvido. Então o que ocorre? Como a gente está mais na seção de pesquisa e desenvolvimento aqui, a gente é responsável por gerar os produtos novos e esses produtos tem uma linha de vida. Então, desde que essa linha de vida foi criada existe esse paralelismo entre os três arquitetos* [enterprise, de soluções e de software, que na empresa é chamado de arquiteto técnico], *porque tem um cara que está olhando tendências de mercado, tem outro cara que está tomando conta do portfólio daquela situação específica e tem um terceiro cara que está preocupado em desenvolver aquilo que é acordado com o arquiteto superior* [o arquiteto enterprise] *que é o cara de serviço. Então é um paralelismo* [nas atividades], *claro que tem, mas assim, o paralelismo dá a ideia de que os três* [arquitetos] *estão trabalhando em conjunto, mas tem pontos de contato****ò*** ï Informante 1, Empresa A.

Estes ñpontos de contatoò mencionados pelo Informante 1, normalmente, são reuniões onde todos os arquitetos, gerentes de projeto, gerentes das unidades de negócio e membros da equipe de *marketing* participam como um ñ*core team*ò para contribuir para a tomada de decisão. Essas reuniões acontecem várias vezes durante o projeto, especialmente em *milestones*. Assim, apesar do fato de cada tipo de arquiteto ter suas atividades específicas e foco em determinada parte do projeto, eles interagem e colaboram bastante entre si.

4.4. Ferramentas e documentação

Outro aspecto interessante a se considerar são os mecanismos que visam apoiar o trabalho de arquitetura. Existem vários mecanismos que visam oferecer apoio às atividades de arquitetura que vão desde metodologias e *frameworks* a ferramentas que prometem apoiar todo o processo de desenvolvimento. Dessa forma, tentou-se identificar quais desses mecanismos as empresas pesquisadas utilizam. Várias abordagens foram identificadas, desde ambientes que visam oferecer apoio centralizado a todo o projeto até o uso de diferentes ferramentas. Assim,

nesta seção serão apresentados os mecanismos utilizados pelas empresas pesquisadas, começando por alguns aspectos da documentação produzida e tratando das ferramentas logo a seguir.

4.4.1. Documentação

Na Empresa C, trabalha-se com alguns documentos pré-definidos pela empresa por fase de projeto, o que inclui o documento de arquitetura. Este documento geralmente é feito pelos arquitetos equivalentes ao arquiteto de soluções com contribuições de outros *stakeholders* e traz informações de tecnologia, uma prova de conceito, a visão lógica e um ñ*cookbook*ò com algumas especificidades de desenvolvimento. Este *cookbook* traz informações técnicas sobre determinados componentes mais importantes do projeto que os desenvolvedores podem usar no dia-a-dia. Entretanto, a presença deste *cookbook* não significa que os arquitetos desta empresa (similares aos arquitetos de soluções da Figura 4.1) também exercem atividades do arquiteto de software da Figura 4.1. Nesta empresa, as atividades do chamado arquiteto de software da Figura 4.1, são exercidas pelos projetistas. O que acontece é que os arquitetos (equivalentes aos arquitetos de solução) descrevem alguns detalhes de componentes mais críticos ou definem algum padrão que deverá ser seguido em todo o desenvolvimento do projeto. Dessa forma, normalmente um documento bastante completo é criado. Nos trechos de entrevista a seguir os informantes desta empresa (ambos similares ao arquiteto de soluções da Figura 4.1) descrevem este documento:

> ñ*A arquitetura em específico, já sai num trabalho um pouco maior, eu, por exemplo, vou pensar na tecnologia, outra pessoa vai pensar na infraestrutura, outra pensar nos testes, outra na gestão de riscos. Tudo isso vai dar origem no que chamamos internamente de* ó*documento de arquitetura*ô*. Nesse documento estão padrões definindo como o nosso sistema se integra com os outros, qual tecnologia iremos utilizar, como nossos sistemas se comunicam internamente, e também como é a organização, do ponto de vista lógico, da nossa solução na questão do desenvolvimento. Isso nós apresentamos pro cliente. Além disso, nesse projeto nós fizemos uma prova de conceito com código fonte*ò ï *Informante 5, Empresa C.*

ñNormalmente ela [arquitetura] *é um documento, ela nasce como um documento (...). Dentro desse mesmo documento a gente mostra primeiro uma visão conceitual depois uma visão lógica. A gente procura mostrar diversos componentes tecnológicas que vai ter na arquitetura, a distribuição disso em camadas, enfim. Depois a gente tem uma parte de decisões de arquitetura que a gente documenta também óah por que isso, por que aquiloô Fala de integrações (...), dependendo do nível né? Porque na primeira versão do documento normalmente fica por aí. Aí na segunda versão, quando a gente já iniciou prova de conceito e tal, a gente amplia essa documentação pra colocar o que é a prova de conceito, os resultados da prova de conceito e algumas instruções de implementação. Que são o que? Instruções que vão ser usadas durante o desenvolvimento pra tarefas comuns dentro da arquitetura que a gente planejou. (...) É basicamente assim que a gente documentaò* ï Informante 4, Empresa C.

Como é possível perceber, esse documento, gerado na Empresa C é bem completo. Ele é bastante usado no início dos projetos, porém os dois informantes desta empresa afirmaram que no decorrer do projeto o mais comum é que os desenvolvedores procurem o projetista (similar ao arquiteto de software da IASA) ou arquiteto (similar ao arquiteto de soluções da IASA) quando querem tirar alguma dúvida, e estes os redirecionam para alguma parte específica do documento que trate da dúvida em questão.

ñNa prática o que acontece? Muitas coisas não se vai direto no documento. Muita coisa do dia a dia e tudo mais acabam indo até o arquiteto consultar, ou até quem é um pouco mais experiente [nos componentes]*, o projetista né?ò* ï Informante 5, Empresa C.

ñNa prática, quando surge dúvida é o arquiteto que é procurado. O que a gente tem tentado fazer é: ólha isso vai ser assim, a gente definiu lá no documento de arquitetura, dá uma olhadaô Os detalhes assim a gente posta lá pra que eles possam encontrarò ï Informante 4, Empresa C.

Este fato (os desenvolvedores procurarem o arquiteto para obter informações) não é exclusividade da Empresa C: todos os informantes afirmaram que na prática isso acontece em suas empresas. Os desenvolvedores geralmente procuram o arquiteto de software quando

possuem alguma dúvida. O arquiteto por sua vez muitas vezes apenas direciona o desenvolvedor, apontando inclusive onde isto está documentado.

Na Empresa A, o primeiro aspecto interessante quanto a documentação trata da interação entre os tipos de arquiteto. Esta empresa possui os 3 tipos diferentes de arquiteto, cada um focando na sua área de atuação. Entretanto, eles não possuem um documento padrão usado para compartilhar as informações entre eles, como o próximo trecho de entrevista ilustra. Essa interação entre arquiteto acontece em reuniões de forma que a documentação gerada é a ata das reuniões.

> ñ*E a gente não tem um, ou pelo menos eu não conheço, nenhum documento que seja oficial de troca de informações entre esses níveis* [os diferentes tipos de arquitetos]. *O que acontece é que existem reuniões específicas, milestones dos projetos, e existe essa interação maior assim. Então, tem uma reunião específica, onde a gente aprova todos os requisitos que vem de marketing, vem da engenharia, vem da onde for, então ali cria um grande ósaco de gatosô dos requisitos. A partir dali é feita uma priorização, gerência de escopo do projeto e aí tem um novo marco no projeto e aí diz assim: óessa versão vai ter esse escopo aquiôò* ï Informante 1, Empresa A.

Outro aspecto interessante relativo a documentação nesta empresa trata dos artefatos que chegam para os arquitetos e os que eles enviam para os outros *stakeholders* (incluindo outros arquitetos). O Informante 1 desta empresa é um arquiteto técnico, que é similar ao arquiteto de software da IASA. No seu trabalho, ele recebe informações de vários *stakeholders* diferentes, as quais vêm usando linguagens diferentes: informações do time de marketing e do cliente vêm em linguagem natural, informações do arquiteto de soluções já vêm em um nível mais técnico, talvez com alguns diagramas de alto nível, se houver alguma informação de um departamento de finanças ela vai trazer uma linguagem típica desta área e assim por diante. Com isso, ele precisa compreender essas várias linguagens para realizar seu trabalho. Sobre esse aspecto este informante declarou o seguinte:

> ñ*É assim, no meu ponto de vista eu acho que esse é um dos objetivos do nosso trabalho: fazer essa tradução dessa quantidade de coisas que acontecem e produzir pro time de desenvolvimento, já de uma forma mais técnica, tirar um pouco dessa linguagem natural.*

Então assim, poderia ter uma padronização de documentação, mas aí eu acho que é muito caso a caso, você não tem como ter um padrão. Mas, aí o que poderia ter é a gente como arquiteto de um componente [arquiteto de software da IASA] *a partir daqui passar pro time de desenvolvimento de uma forma mais padronizada, pra que isso facilitasse o entendimento e pra que tirasse um pouco a carga do arquiteto de ter que fazer uma explicação mais detalhada do que significa esse requisito. Então, eu acho que nesse ponto cria uma contribuição maior do que conseguir fazer com que o mundo inteiro que fala várias línguas estivesse utilizando um mesmo artefato. O cara de marketing tem responsabilidades específicas, o cara de finanças tem responsabilidades específicas. Então, assim é difícil fazer esses caras conversarem com a mesma cor. Então, tem que ter um cara que entenda um pouquinho de cada um e tente traduzir pra fazer, mas a partir daqui tem que chegar redondo pro time de desenvolvimento pra que tu tenhas um entendimento melhor do que é aquilo que tu vais fazer e especialmente pra que tu não construas uma coisa equivocada. Tipo assim: óah, eu entendi isso e a gente construiu exatamente o que a gente entendeu só que deu errado. Agora a gente vai fazer tudo de novoôò ï Informante 1, Empresa A.*

No trecho de entrevista apresentado acima, um aspecto interessante a ser abordado é mencionado: a tradução, o que será descrito na seção 4.6 deste capítulo. Neste ponto, o intuito é ilustrar o uso da documentação pelas empresas e o que, na opinião dos arquitetos, poderia ser feito ou o que eles consideram importante. Nesse sentido, outro aspecto interessante relacionado ao trecho de entrevista acima é a passagem dos requisitos que serão desenvolvidos para o time de desenvolvimento. O Informante 1 relatou que no projeto atual em que ele está trabalhando eles dedicaram um tempo maior para criar um *template* de padronização dos requisitos para tentar melhorar o entendimento dos mesmos por parte dos desenvolvedores, além de adotarem uma ferramenta de apoio ao gerenciamento dos requisitos que será discutida posteriormente. O trecho de entrevista a seguir trata exatamente disso e das consequências da adoção deste *template*.

ñ*No projeto que a gente está trabalhando hoje está um pouco melhor* [a passagem das informações para os desenvolvedores], *porque a gente dedicou um certo tempo em tentar padronizar os requisitos e criar um template pros requisitos. E a gente começou a utilizar*

*uma ferramenta de apoio a gerenciamento de requisitos, pra tentar melhorar o gerenciamento e esse template pra tentar melhorar o entendimento dos requisitos. A gente já conseguiu ter algumas vantagens com relação a isso, mas a gente está longe ainda do estado que a gente gostaria de estar. Hoje, o time de desenvolvimento ainda tem que sofrer bastante pra definir melhor as coisas quando chegam assim. Isso é uma coisa que realmente ainda desejo: (...) as coisas chegarem assim mais certas, mais definidas pro time de desenvolvimento, pra gente se ocupar só em desenvolver e não em compreender a lógica do projeto***ò*** ï Informante 1, Empresa A.

Dessa forma, um aspecto importante trata dessa passagem de informações para os desenvolvedores. Entretanto, qualquer padronização disto pode ao final não obter o resultado desejado. As equipes de desenvolvimento são muito diferentes umas das outras: sua maturidade, interação, expertise, etc. Com isso, é importante que o arquiteto conheça sua equipe para poder repassar as informações da melhor maneira possível para ela, de forma que se gaste menos tempo tentando compreender os requisitos. O Informante 14, também um arquiteto técnico da Empresa A (similar ao arquiteto de software da IASA), afirma que não adianta padronizar, que a melhor abordagem é adaptar a documentação de acordo com a equipe, como ele relata nos seguintes trechos de entrevista.

ñ*Eu acho que isso* [a documentação] *depende da interação com o time, então tem que ser desenvolvido de acordo com cada equipe. Não adianta tu padronizares aqui pra a melhor documentação que eu consigo fazer: vai ter todos os diagramas de sequência pra dizer exatamente como é que tudo tem que funcionar. Só que aí provavelmente tu vais estar perdendo muito tempo pra fazer aquilo que não vais estar poupando do outro lado* [desenvolvimento]. *Então, tem que ir melhorando aos pouquinhos e vendo quão específico o arquiteto pode ser e quanto isso vai estar ajudando a equipe de desenvolvimento, dependendo da maturidade da equipe de desenvolvimento, então tem que ir ajustando isso aos pouquinhos, não tem como chegar* ***ó****ah, vou fazer assim em todos os projetos****ô*** *e* [achar que] *vai estar correto pra todos os projetos****ò*** ï Informante 14, Empresa A.

ñ*A única coisa que eu tenho certeza que funciona é ir adaptando de acordo com o projeto. Tem projetos que funcionam muito bem com uma historinha de três ou quatro linhas*

*colada num quadro branco. Tem projetos que isso só funciona se tiver um documento de várias páginas explicando exatamente como é que vai funcionar. Então, isso varia muito de acordo com cada projeto***ò** ï Informante 14, Empresa A.

Na Empresa A, como descrito nos trechos anteriores, os informantes alertam para a necessidade de adaptar a documentação de acordo com a equipe de desenvolvimento, uma vez que eles são arquitetos técnicos e trabalham diretamente com desenvolvedores. Na Empresa D, o informante 17 também relatou que a documentação depende do projeto, mas no geral o documento não é algo rígido e o arquiteto o desenvolve da maneira que achar apropriada:

ñ**Esse esboço arquitetural não é um documento rígido, certo? (...) Isso já vem do background de cada arquiteto, ele vai desenvolver esse esboço arquitetural usando o que quiser, eu, por exemplo, gosto de usar o UML, certo? Mas, não é nenhum impeditivo que a gente utilize nesse plano arquitetural só o quadro branco, diagramas de caixas e fluxos normaisò** ï Informante 17, Empresa D.

Na Empresa E também ocorre algo similar: a documentação depende do projeto, mas no geral ela é bem informal. Entretanto, geralmente um documento relativamente padrão é produzido para todos os projetos. Este é um artefato simples e em alto nível que é descrito pelo Informante 7 no seguinte trecho de entrevista:

ñ**Basicamente é um fluxograma onde eu tenho os comportamentos do sistema. E são caixas macros. Cada caixinha dessas depois é aberta, cada caixinha cinza na verdade (...), é aberta num outro diagrama parecido com esse que vai detalhando a lógica de funcionamento do programa até que chega um ponto que diz realmente a regra de negócio que vai ser reaplicada naquele ponto, nós já detalhamos dessa maneira. E junto com isso, caso alguma coisa não fique muito clara no desenho, apesar da gente ter todos esses balões aqui, a gente coloca num documento, e a gente usa "wiki", alguma definição melhor: **ó**olha! isso aqui tem que ser assim mas tem que se lembrar disso**ô** Ponto. Essa pra mim é uma documentação que é fácil de fazer, ela é fácil de entender e é muito útilò** ï Informante 7, Empresa E.

Nesta empresa, o Informante 7 também afirmou que documentação não é algo supervalorizado por eles, eles só produzem o ñnecessárioò, como descrito no trecho de entrevista anterior. Algo similar a isso também ocorre nas Empresas G e D, onde há documentação, mas a maioria das informações é repassada através de reuniões, como pode ser constatado com os trechos de entrevista a seguir. No primeiro, o Informante 9 da Empresa F cita as reuniões como forma de troca de informações e também cita o código, que é a principal documentação desta empresa. No segundo trecho de entrevistas, o Informante 17 da Empresa E cita um *framework* de arquitetura relatando que eles costumam usar alguns pontos deste *framework*, mas relata que na prática o arquiteto é a fonte de informações da equipe. Na verdade, nos dois trechos de entrevista este aspecto se destaca: o arquiteto como fonte de informações da arquitetura.

ñ*Tem a Ferramenta F* [uma wiki] *lá que eles chamam, que sempre que tem tempo alguém documenta. Então, tudo não está documentado lá, porque muita coisa está na cabeça de um dos nossos arquitetos. Então, eu tive uma solução lá, eu coloco, e passo pra eles numa reunião. Mas, se eu documentei, isso não me é cobrado. Então, a gente tenta documentar tudo, mas nem sempre isso acontece.*[a passagem das informações para outros stakeholders é feita por] *Javadoc. E nós temos reuniões - dentro da nossa equipe Java a gente tem reuniões mensais. Nessas reuniões a gente coloca tanto problemas, quanto soluções novas. Então, isso é passado pra eles por isso e algumas coisas estão na web lá, interno****ò*** ï Informante 9, Empresa F.

ñ*Na verdade o processo de arquitetura, ele é guiado por linhas gerais, certo? A gente não utiliza nenhum* ***ó****framework****ô*** *de arquitetura como, por exemplo, o TOGAF, certo? Isso é até uma coisa que internamente entre os arquitetos, está havendo aderências ao TOGAF aos nossos processos. (...) A gente tem lá as etapas de arquitetura, tem as etapas de refinamentos, mas o quê que acontece? Na prática, esse conhecimento é do arquiteto em questão. A gente só sabe que, a gente tem que alocar em determinado estágio do projeto, o expertise de um arquiteto****ò ï*** Informante 17, Empresa D.

4.4.2. Ferramentas

Um aspecto observado durante a pesquisa foi o uso de ferramentas pelos diferentes arquitetos de TI, incluindo arquitetos de software. Através da análise das entrevistas com os informantes, notou-se que as organizações tendem a usar um misto de ferramentas, geralmente simples. Além disso, como a seção anterior ilustrou, a documentação produzida tende não ser muito formal, o que reforça o uso de ferramentas. Esta seção trata das ferramentas usadas pelas empresas identificando, através de trechos de entrevistas, alguns aspectos aos quais elas não atendem ou algumas dificuldades relacionadas enfrentadas pelos arquitetos de TI.

O Informante 1 da Empresa A relatou que a transferência de informações dos arquitetos (similares ao de software da IASA) para os desenvolvedores deveria acontecer de uma forma mais padronizada para que estes últimos pudessem se concentrar apenas no desenvolvimento. Isso diminuiria um pouco do peso do arquiteto ter que explicar em detalhes o que os requisitos significam para o time de desenvolvimento. Por isso, no projeto atual em que ele trabalha eles criaram um *template* para os requisitos, que visa melhorar a compreensão deles, e passaram a usar a Ferramenta A para gerenciar melhor os mesmos requisitos.

Além disso, a Empresa A é uma multinacional e seus diferentes tipos de arquitetos não são colocalizados. Por isso, é muito importante para esta empresa ter mecanismos que apoiem as atividades dos arquitetos, especialmente considerando suas interconexões e dependências, levando em consideração a distância geográfica entre eles. A Ferramenta A, utilizada pela empresa, é uma ferramenta de gerência de testes que eles usam para gerência dos requisitos, especialmente por favorecer revisões *offline* dos mesmos. Estas revisões são importantes para que as reuniões sejam mais produtivas, pois as pessoas podem revisar (ler) os materiais e então ir para as reuniões somente para discutir os assuntos. Isto é importante porque, devido à distância, não é simples reunir os *stakeholders* em reuniões, especialmente quando os fusos-horário são muito diferentes. O Informante 1 descreve brevemente o uso da Ferramenta A a seguir:

> ***ñ****A gente está usando aqui a Ferramenta A.(...) Ela impõe um fluxograma de trabalho que eu vou usar. (...) Então é o seguinte, tem já um processo pra controlar mudanças de requisitos, mudanças no projeto e essa ferramenta nos ajuda nisso. Ajuda você a criar os requisitos lá dentro e a gente consegue facilitar a revisão dos requisitos por times remotos.*

E esse é um grande ponto do problema, porque nos nossos projetos que a gente trabalha hoje, a gente trabalha com pessoas em várias localidades e conseguir que todos eles estejam num mesmo momento pra fazer uma revisão e por telefone não é totalmente produtivo. Então, o ideal é ter um pouco mais de trabalho offline que eu consiga realizar e chegar pra reunião pra tirar dúvida **ï** *Informante 1, Empresa A.*

Entretanto, apesar da utilização da Ferramenta A, a questão de requisitos, especialmente a revisão deles, continua sendo um problema para o Informante 1. A próxima seção irá descrever esse problema e outros relacionados a requisitos que os informantes relataram.

A Empresa C adota outra ferramenta (chamada de Ferramenta C) que armazena e provê rastreabilidade entre os diferentes tipos de documentação. Esta ferramenta consiste de um ambiente de modelagem que afirma apoiar todo o ciclo de vida de desenvolvimento. Os trechos de entrevista a seguir descrevem o uso da Ferramenta C nesta empresa:

ñ*A gente vem começando a usar muito aqui também ferramentas que integram tudo, que nem a Ferramenta C, que aí eu consigo deixar os diagramas lá, consigo fazer a documentação técnica ligada aos requisitos toda ali dentro né? Então, nessa parte de especificação já das funcionalidades e tal a gente trabalha dentro de uma ferramenta, a gente trabalha assim. Embora o documento continue existindo, o documento é a referência né?***ò** ï Informante 4, Empresa C

ñ*Mas, a ideia é que toda parte de documentação do projeto ficasse dentro da Ferramenta C, cada um com as suas seções, e conforme o projeto fosse andando, fosse crescendo também os tipos de diagrama, o modelo crescendo e toda ela tendo essa ligação, porque nos faz crer que o desenvolvedor lá na classe dele, ele consiga ir subindo até chegar por exemplo num requisito não funcional que está sendo atendido por aquela classe do sistema, ou do componente***ò** ï Informante 5, Empresa C.

Entretanto, o Informante 16 da mesma empresa cita que eles estão adotando uma prática de usar a mesma Ferramenta C para fazer engenharia reversa do código e gerar a

documentação com base nela. O caminho tradicional de projetar primeiro para depois codificar é realizado apenas no caso de componentes mais críticos que precisam dessa definição antes.

> *ñAgora a última prática que a gente tem adotado, é utilizar a Ferramenta C, mas mais pra gerar engenharia reversa do que já foi codificado, do que seguir o caminho mais tradicional, que manda a literatura, que é primeiro projeta, aí depois em cima do que foi projetado é gerado o código. Isso é bem mais difícil da gente enxergar aqui dentro. É que são poucos projetos ou poucos componentes dentro do projeto, que se tem esse cuidado, de fazer o projeto de software mesmo, uma ferramenta de modelagem e tudo mais. Via de regra, é uma coisa, vamos chamar de mais ágil, pra manter a elegância, é uma coisa mais ágil, ou seja, pega um papel, desenha ali, conversa com o desenvolvedor ou enfim, já mete a mão na massa. Depois, se precisar, se o cliente pedir ó ah eu quero uma documentação do ponto de vista técnicoô etc, aí gera uma reversa.ò - Informante 16, Empresa C.*

Diferentemente das abordagens anteriores, a Empresa I customizou um workflow de trabalho implementado em um servidor web comercial, chamado aqui genericamente de Ferramenta I. Na mesma linha de customização de ferramentas, a Empresa D customizou a Ferramenta D, que é uma ferramenta web para gerência de projetos e *bugtracking*. Ambas as empresas customizaram as ferramentas para atender suas necessidades relacionadas a requisitos. Na Empresa I a customização foi feita para que os clientes pudessem cadastrar os requisitos na ferramenta e a partir daí seguisse o workflow de desenvolvimento da empresa, passando pelos outros atores como *product owner*, arquiteto, desenvolvedores, etc. Na Empresa D a customização foi feita no sistema de *trackers* da ferramenta de gerência de projetos visando o uso para atividades de arquitetura entre outras. Os seguintes trechos de entrevista relatam essas customizações:

> *ñBom, a gente tem um workflow de trabalho que na verdade foi customizado e implementado dentro de uma ferramenta que a gente utiliza, que é a Ferramenta I. Dentro dessa ferramenta tem a possibilidade de customizar o workflow conforme a nossa necessidade. Então, lá inicialmente o cliente cadastra os requisitos, esse projeto que eu*

estou trabalhando é bem focado em CR, a gente vai lá e cadastra uma solicitação de mudança **ï** *informante 21, Empresa I.*

ñ*Em termos de artefatos a gente utiliza bastante a congregação de controle de demandas, certo? A gente customiza o controle de demandas pra representar requisitos, pra que depois a gente consiga fazer uma referência cruzada. Então, por exemplo: a gente tem uma ferramenta de gerenciamento de projeto que se chama Ferramenta D. Ela tem integrado gerenciamento de riscos, wiki e uma série de ferramentas de gestão de projeto, certo? Então o que a gente faz? Customiza o que a gente chama de trackers, semelhante a ferramentas de bugtracking. A gente tem trackers pra aspectos da arquitetura, pra requisitos de arquitetura. A gente atende, aloca em interações, tá? E a gente pega essa documentação e põe no wiki. O nosso artefato é o wiki***ò ï** *Informante 17, Empresa D.*

Outro aspecto interessante identificado está relacionado com a alta utilização de wikis. A maioria das empresas observadas tende a usar documentos comuns e wikis para criar e manter as ñdiferentes arquiteturasò (especialmente a arquitetura de software) e a documentação associada a elas. Mais especificamente, as Empresas E, D e G utilizam wikis e a Empresa I, como descrito acima, usa uma ferramenta que se assemelha a uma wiki. Os trechos de entrevista abaixo ilustram o uso de wikis pelas empresas.

ñ*Informante: Tem a Ferramenta F* [uma wiki] *lá que eles chamam, que sempre que tem tempo alguém documenta. (...)*

Pesquisador: E como é feita a passagem das informações para outros stakeholders?

Informante: *Javadoc. E nós temos reuniões - dentro da nossa equipe Java a gente tem reuniões mensais. Nessas reuniões a gente coloca tanto problemas, quanto soluções novas. Então, isso é passado pra eles por isso e algumas coisas estão na web lá, interno* [na Ferramenta F]**ò**ï Informante 9, Empresa F.

ñ*Odeio ficar preso em ferramenta: a gente usa o wiki, a gente usa o powerpoint e a gente usa a Ferramenta E* [ferramenta de diagramação]. *A gente usa o quadro branco com foto também que fica super bom pra documentar muita coisa***ò** ï Informante 7, Empresa E.

*ñA gente trabalha muito usando wiki, certo? Então na verdade o que acontece? Tem essa priorização dos pontos de atenção do projeto. A gente faz os modelos pontuais, mas o que vai nortear vai ser uma documentação do wiki, com alguns aspectos que a gente põe lá***ò** ï Informante 17, Empresa D.

*ñO nosso artefato é o wiki***ò ï** *Informante 17, Empresa D.*

4.5. Requisitos

O trabalho dos arquitetos está muito ligado aos requisitos tanto funcionais quanto não funcionais (principalmente estes últimos). Todos os informantes declararam de alguma forma que também trabalhavam no levantamento de requisitos, com exceção daqueles que não interagem com os clientes, como na Empresa F. Entretanto, os arquitetos desta empresa declararam que gostariam de ter essa interação, ou ao menos uma interação maior com os próprios analistas, especialmente no momento da definição do negócio e definição do escopo.

O trabalho dos arquitetos em relação aos requisitos (especialmente os não funcionais) é bem frequente, definido até mesmo como ñfundamentalò por alguns deles. Este trabalho envolve desde a elicitação, especialmente no que tange levantar requisitos não funcionais a partir das necessidades do cliente, até o repasse dos mesmos para os desenvolvedores. Os trechos de entrevista a seguir ilustram esse trabalho dos vários tipos de arquitetos com requisitos.

*ñNós olhamos para a Arquitetura desde a iniciação do projeto, onde nós temos oportunidade de olhar o que existe, mas de fato o trabalho da arquitetura começa na elaboração do projeto, que é antes da construção. Nós vamos ter uma fase ali que nós vamos começar a definir padrões, talvez até a especificar alguns requisitos mais críticos , aprovar isso para que quando chegue na construção já tenhamos todo o material possível para que toda a equipe de desenvolvimento consiga trabalhar da mesma maneira. Então temos essas etapas bem definidas***ò** ï Informante 5, Empresa C.

ñComo arquiteto eu trabalho bastante voltado a requisitos, principalmente requisitos não funcionais tá? Então por exemplo, a primeira coisa que eu foco pra ter o entendimento do

*que eu devo fazer numa nova atribuição é perguntar qual a carga que o sistema vai ser submetido. Então, geralmente eu posso trabalhar com levantamento de requisitos funcionais, certo? Mas, eu sempre vou ficar focado na análise de requisitos não funcionais. Eu trabalho bastante no pré-venda aqui. Então sempre que há uma nova demanda aqui, eu sou envolvido, aí eu trabalho em conjunto com os analistas de requisitos pra tentar estruturar uma visão geral do sistema bem nessa fase inicial e tentar achar possíveis pontos de atenção no sistema. Ah, o sistema é distribuído? o sistema vai precisar de ter um tempo de vida longo? ou vai ter um tempo de vida curto? quantos usuários? quantos sites vão estar envolvidos no sistema? (...). Então, na verdade a partir da visão geral eu tento encontrar certos pontos de atenção principalmente focado em requisitos não funcionais, pra tentar abordar e montar uma estratégia de como se lidar com esses tipos de pontos aí. Então por exemplo, o mais comum é performance e escalabilidade, certo? Mas também pode ter manutenção. Uma série de coisas***ò*** ï Informante 17, Empresa D.

ñ*A primeira coisa é levantamento de requisitos. É o fundamental. A gente trabalha como uma ponte entre o time de marketing e o time técnico. Uma das atividades básicas é o levantamento de requisitos e garantir que a tradução foi feita correta e o trabalho regular de ver onde se encaixa porque todos os produtos seguem uma linha e normalmente eles estão inclusos em algum fluxo de execução de um datacenter típico****ò*** ï Informante 1, Empresa A.

ñ*Eu vejo o arquiteto de software mais do lado da parte de desenvolvimento, mas sempre ouvindo o cliente também. Porque tem coisas que o cliente fala que não são requisitos, mas que de alguma maneira podem influenciar o desenvolvimento****ò*** ï Informante 7, Empresa E.

ñ*A arquitetura trabalha junto* [com os analistas de negócio e de sistemas] *eventualmente decidindo se é necessário fazer algum tipo de alteração na arquitetura ou se a arquitetura atual comporta o desenvolvimento do requisito****ò*** ï Informante 21, Empresa I.

Como pode ser percebido através dos trechos de entrevista anteriores, o trabalho dos arquitetos está muito ligado aos requisitos (especialmente os não funcionais), não importa o

tipo de arquiteto, uma vez que os requisitos influenciam diretamente na arquitetura. A presença do arquiteto, especialmente na definição do contrato ou de escopo, é muito importante pois ele pode capturar requisitos que o cliente não saberia descrever ou que podem passar despercebidos pelos analistas e outros *stakeholders*. Como o Informante 7 da Empresa E citou, existem aspectos que o cliente comenta que não são requisitos ,mas que influenciam no desenvolvimento da arquitetura. O Informante 1, da Empresa A, citou o exemplo de sistemas que devem permanecer sempre no ar. Se o cliente for mais acostumado com os termos, ele pode definir o mínimo de disponibilidade que o sistema deve ter. Entretanto, se ele não tiver tanto contato com essas nomenclaturas, o arquiteto vai ter que reconhecer e interpretar o que ele fala para poder obter as informações como as descritas pelo Informante 1 no trecho de entrevista abaixo:

> ***ñ****Em storage duas coisas são fundamentais: primeiro a disponibilidade dos dados. Quanto maior for a disponibilidade do dado maior é o clamor por alto nível daquele produto de storage. Se tu queres chegar no nível de um mercado tipo uma bolsa de valores ou um mercado onde server é non-stop, qualquer segundo de downtime é de altíssimo preço, tu tens que ter um sistema de disponibilidade de cinco noves. Já ouviram falar nisso? É 99,999% do tempo disponível, o que dá em torno de mais ou menos um downtime de 5 minutos/ano. Então, eu não posso, em contrato, passar isso. Não pode o dado ficar indisponível mais que 5 minutos no ano. Isso é inaceitável. Então o produto de enterprise pra esse tipo de companhia precisa atingir esse requisito****ò ï*** *Informante 1, Empresa A.*

Essas informações são muito importantes para que o produto construído realmente atenda as necessidades do cliente. Caso contrário vários problemas podem surgir. É importante ressaltar que isso não significa que os arquitetos exerçam as atividades dos analistas, mas que eles podem colaborar com o trabalho dos analistas, especialmente no que se refere a requisitos não funcionais, que influenciam diretamente na arquitetura de um sistema.

Como mencionado anteriormente, os informantes relataram ter problemas com relação a requisitos, mesmo aqueles que declararam que o trabalho com requisitos faz parte das suas atividades. Tais problemas se relacionam principalmente em um requisito ser desenvolvido como foi especificado, mas no final não representar o que o cliente realmente queria. Por

exemplo, na Empresa F os arquitetos não tem contato com o cliente, nem mesmo nas reuniões de definição do contrato e escopo e mesmo o contato com os analistas, que nesse caso são os responsáveis pela interação com os clientes, é bastante reduzido. Com isso, o Informante 9 afirma que problemas de entendimento de requisitos (como representado na Figura 4.9) acabam surgindo, como ele explica no trecho de entrevista a seguir:

> ñ*Nós não estamos lá na parte quando está começando o projeto. A gente já pega o projeto com ele mais ou menos definido. Está mais ou menos especificado. (...) não são todos os projetos que nós temos reuniões com os analistas. Então, eles só nos enviam a tarefa e a gente desenvolve em cima disso. (...) Eu acho que seria essencial ter sempre* [comunicação com analista e/ou cliente] *porque ia diminuir muito alguns problemas que eu coloco, tipo: a gente desenvolve alguma coisa, desenvolve uma solução que o analista montou pra nós e três dias depois começa a cair muita informação em cima dela e ela é derrubada, ela fica toda hora caindo, ela prende o nosso banco* [de dados]. *Como a gente depende muito do banco* [de dados] *pro negócio (...), a gente não poderia ter só desenvolvido e pronto. A gente teria que ter tido uma reunião com eles: ó ah, isso aqui não pode ficar aqui, não vai dar certo no futuro, é uma carga muito grande, isso vai travar ô. A comunicação seria essencial lá no início, o que não aconteceu. E a gente para o projeto, vê se o cliente que está mandando essa gama de informação é muito grande e tal, então esse cliente a gente vai ter que deixar pra fazer de madrugada. E isso não pode, o cliente não quer fazer isso de madrugada, ele quer ter instantâneo a informação. E isso foi por que? Porque não aconteceu uma comunicação do arquiteto pra ver se a solução realmente era suficiente* ï Informante 9, Empresa F.

Já os informantes da Empresa A se queixaram quanto a revisões dos requisitos. Para eles, se as pessoas costumassem revisar os requisitos no decorrer do projeto esse problema poderia ser amenizado. Entretanto, não é isso que acontece. No caso desta empresa as revisões são ainda mais importantes devido a distribuição geográfica das equipes. A ideia é que as pessoas revisassem os requisitos antes das reuniões para que o tempo fosse melhor aproveitado com dúvidas, uma vez que conseguir marcar reuniões com times distribuídos é sempre mais complicado devido a questões como fuso-horário diferente. O Informante

1inclusive cita que as ferramentas poderiam ser mais atrativas para motivar as pessoas a fazer isso. Nas palavras dele:

> *ñEu falei sobre a revisão dos documentos. As pessoas não gostam de ler, tem gente que não gosta coisas que elas não entendem e esperar que todo mundo vá fazer uma revisão offline, chegar na reunião marcada e só ter os pontos de dúvida específicos, isso não ocorreò* ï Informante 1, Empresa A.

> *Uma coisa que eu gostaria de ter era uma ferramenta que a gente pudesse submeter os requisitos (...) pra que nós pudéssemos ir detalhando, detalhando, detalhando e ficou assim ó é isso aqui que a gente vai criarô e que as pessoas pudessem aprovar assim: órealmente, isso aqui foi levantado por mim e eu consegui aprovarâ. Não é que a gente não tenha uma ferramenta, a gente tem, pode ser que ela permita isso, a Ferramenta A permite isso, só que ela não é boa o suficiente que motive as pessoas a ir nela fazer issoò*ï Informante 1, Empresa A.

Os Informantes 14 e 15 complementaram, nos trechos de entrevista a seguir, esse aspecto das revisões comparando os problemas que essa falta de revisão pode causar com a Figura 4.9 (apresentada após os trechos de entrevistas), ressaltando a gravidade desses problemas.

> *ñAquilo ali, assim, não é um balanço e uma árvore que nos custam, é muito dinheiro, muito dinheiro. Deixar pra descobrir que a gente fez a coisa certa, exatamente como tinha sido escrito, gastamos um tempão testando e a gente chegou no final achando que a gente era super hiper ultra bom e descobrir que tudo que a gente fez na verdade não resolvia o problema original, aquilo que se pensava em fazerò* ï Informante 14, Empresa A.

> *ñA falta de uma revisão pode botar 6 meses de trabalho de 12h a 14h* [por dia] *no lixoò* ï Informante 15, Empresa A.

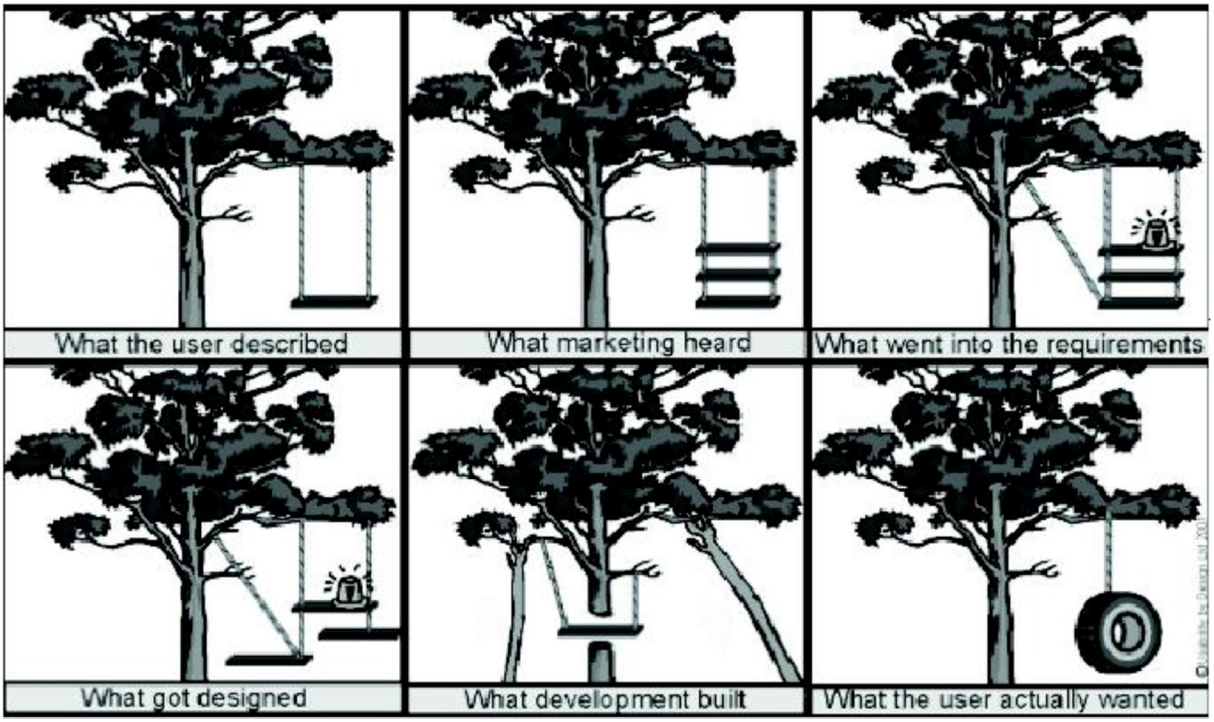

Figura 4.2 Representação do problema com requisitos

Como se pode perceber, problemas com requisitos são graves e podem custar caro para as empresas. Eles podem acontecer mesmo quando os arquitetos (incluindo todos os tipos) participam de todo o projeto. Quando isso não acontece, a probabilidade de problemas como os citados ocorrer pode ser maior. Dessa forma, os arquitetos não podem separar suas atividades dos requisitos e devem colaborar mais frequentemente com analistas, além de, de preferência, interagir também com clientes para capturar informações que podem nem ser requisitos, mas que podem ser fundamentais para o sucesso do sistema produzido. O Informante 17 resume essa necessidade nos seguintes trechos de entrevista:

> ***ñ****Porque normalmente o que a indústria vê? Ela vê o arquiteto de software como um desenvolvedor pós-sênior, e isso não é verdade. Então, o desenvolvedor pós-sênior, ele vai estar focado na tecnologia. O arquiteto, ele tem que estar focado na tecnologia, nos requisitos não funcionais, e nos aspectos sócio-políticos do sistema****ò*** ï Informante 17, Empresa D.

> ***ñ****Por isso que eu vejo que a arquitetura, eu vejo ela como sendo indissociável da engenharia de requisitos****ò*** ï Informante 17, Empresa D.

4.6. Tradução de informações

Através dos dados extraídos das entrevistas e apresentados nas seções 4.2 e 4.3 (e respectivas subseções) percebe-se que as divisões de atividades de arquitetura e as consequentes interações relacionadas a elas contribuem para um fluxo de informações dentro do projeto. Entretanto, analisando este fluxo de informações, percebe-se que, através dele, acontece a ñtraduçãoò das necessidades do cliente para termos técnicos que podem ser implementados pelos desenvolvedores. Explicando melhor: os arquitetos *enterprise* trabalham mais em nível organizacional, definindo padrões de TI que serão adotados nos projetos da empresa e trabalhando na definição de contratos com clientes externos, ou de novos projetos internos ou requisitados pela área de *marketing* (também chamada de área de produto) da empresa. Para isso, eles interagem com *stakeholders* como a alta gerência, *marketing* e clientes e lidam com informações em linguagem natural e recheadas de termos e variáveis organizacionais e de mercado (por exemplo: custo, ROI, aspectos financeiros, tendências de mercado, análises de concorrência, etc). Portanto, eles precisam compreender estes aspectos para identificar aqueles que são importantes e influenciam a arquitetura de TI e as arquiteturas de software dos sistemas produzidos. Ao fazer isso, eles agregam partes importantes das informações dos diferentes *stakeholders* e também agregam a estas suas próprias informações. Durante esse ñprocessoò eles traduzem essas informações de forma a passar um novo tipo de informação aos outros *stakeholders*. Essas novas informações podem ser consideradas como requisitos organizacionais e já estão em um formato de linguagem um pouco menos natural.

Os arquitetos de soluções recebem esses requisitos organizacionais, mas recebem também informações de clientes e analistas (dessa vez relacionadas aos requisitos funcionais e não funcionais do sistema que será produzido). Mais uma vez, estas são informações em formatos diferentes, as quais o arquiteto de soluções agregará também com as suas próprias informações, criando assim um novo tipo de informação com um caráter mais próximo da linguagem técnica de desenvolvimento. Este novo tipo de informação é constituído principalmente da visão geral da arquitetura e já acumula as informações organizacionais com as quais o arquiteto *enterprise* trabalhou. Este novo tipo de informação pode ainda ser dividido para assim ser repassado aos *stakeholders* que trabalharão com ele, incluindo os arquitetos de software.

Por fim, o arquiteto de software recebe as informações relativas aos módulos que a sua equipe desenvolverá. Tais informações estão em uma linguagem intermediária, geralmente

uma linguagem de modelagem com alguns aspectos técnicos. Com isso, os arquitetos de software vão adicionar as suas próprias informações para refinar os módulos de forma a deixá-los com uma linguagem de nível técnico que possa ser utilizada pelos desenvolvedores para a construção do software. Assim, a necessidade do cliente (referindo-se ao projeto como um todo e não necessariamente aos requisitos) foi ñtraduzidaò pelos diversos papéis até que alcançasse os desenvolvedores. O seguinte trecho de entrevista ilustra esse ñprocessoò:

> ***ñ****Tu tens o enterprise architect lá que vai ver o que o cliente precisa, vai falar com o pessoal de marketing, vai juntar todas as suas informações e vai ter uma ideia de o que vai ser vendido pro cliente e isso passa pra um ou mais solution architect e esse cara vai ver, algumas vezes, quais componentes prontos que eu tenho aqui na empresa que a gente pode utilizar, vai fazer a divisão entre vários projetos ali dentro. Aí aqui tu tens uma divisão de informações. Na maior parte das vezes não precisa chegar todas as informações que o enterprise architect tinha lá em cima até cada um dos componentes, então nesse cascateamento tu diminuis a quantidade de informações, mas aumenta o nível de detalhes a cada um****ò*** ï Informante 14, Empresa A.

CAPÍTULO 5

5. Discussão

5.1. Visão geral

Os resultados abordados no capítulo anterior apontaram a tendência de as empresas pesquisadas dividirem as atividades de arquitetura em diferentes tipos de arquitetos, ressaltando as interações entre eles e com outros *stakeholders* das empresas. Essa divisão pode ser tanto formalizada pela organização quanto feita diretamente pelos próprios arquitetos. Além disso, aspectos relativos à documentação, ferramentas e requisitos foram apresentados. Tais aspectos estão muito relacionados às atividades dos arquitetos e, portanto, são importantes para o trabalho dos mesmos e para a forma que as empresas realizam suas atividades de arquitetura.

Este capítulo apresenta uma discussão sobre os resultados do capítulo anterior, procurando analisar e relacionar todos os aspectos identificados e descritos anteriormente. Tal discussão visa compreender como os arquitetos de TI realizam as atividades de arquitetura na indústria, como informando no primeiro capítulo. Ela visa também discutir as implicações destes resultados para a pesquisa e ensino da arquitetura de software na academia.

Primeiramente um *framework* de avaliação e compartilhamento de conhecimento será descrito, pois os resultados serão analisados usando o mesmo. Este *framework* é descrito no trabalho de Paul Carlile (2003) e discute como o conhecimento é compartilhado através de três tipos de fronteiras: sintática, semântica e pragmática. Tais fronteiras serão explicadas e o papel dos diferentes tipos de arquitetos no compartilhamento de conhecimento através delas será identificado. A partir dessa análise, também será discutida a influência dos aspectos de documentação, ferramentas e requisitos e como a indústria vem lidando com eles, destacando por fim algumas hipóteses decorrentes deste trabalho.

5.2. Compartilhamento de conhecimento através de fronteiras

Para analisar os dados coletados e apresentados no capítulo anterior, utilizou-se o *framework* desenvolvido por Carlile (2003). Este explorou as fronteiras de conhecimento e os processos utilizados para que o conhecimento seja compartilhado através delas. Para isso, Carlile (2003) desenvolveu um *framework* para unificar visões diferentes da área de gerência de conhecimento.

Antes de apresentar o *framework* é importante explicar a ideia de fronteiras de conhecimento, as quais são criadas quando existem domínios especializados de conhecimento e os indivíduos destes domínios precisam interagir e trocar informações (Carlile e Rebentisch, 2003). A Figura 5.1 abaixo traz um esquema de um exemplo de uma fronteira de conhecimento descrita por Carlile (2003). Nela a situação é a seguinte: em uma empresa de fabricação de carros existia uma equipe especializada em motores e outra especializada no estilo ou *design* dos carros, cada uma possuindo conhecimento específico sobre sua área. Em um determinado momento a equipe de motor desenvolveu um novo motor mais potente e econômico e resolveu utilizá-lo no próximo modelo de carro da empresa. Ao mesmo tempo, a equipe de estilo desenvolveu um novo *design* com uma aparência mais aerodinâmica para o carro e também resolveu usá-lo no próximo modelo. Aparentemente, a decisão de uma equipe não influencia a decisão da outra equipe. Entretanto, ao unir as duas decisões percebeu-se que elas eram incompatíveis, pois enquanto o novo motor apresentava dimensões maiores, o novo estilo apresentava um menor compartimento para o motor. Dessa forma, existia uma fronteira entre as duas equipes através da qual o conhecimento de ambas deveria ser compartilhado para que problemas assim não ocorressem.

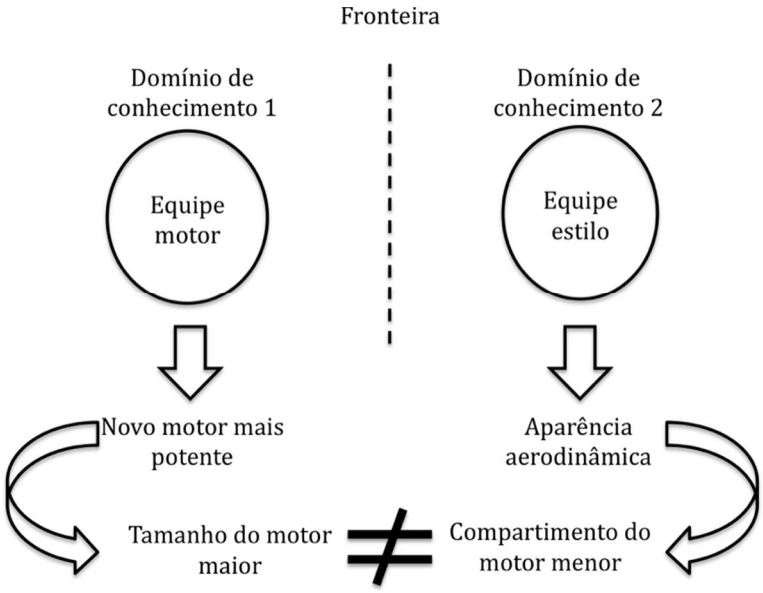

Figure 5.1 Exemplo de fronteira de conhecimento

O *framework* de Carlile (2003) identifica os tipos de processos de compartilhamento de conhecimento que são necessários em cada tipo de fronteira. Para ser mais específico, os processos de compartilhamento de conhecimento discutidos por Carlile (2003) são: transferência, tradução e transformação de conhecimento. Note que fronteiras de

conhecimento mais complexas exigem processos de compartilhamento de conhecimento também mais complexos. A Figura 5.2 a seguir apresenta o *framework* onde os três tipos principais de fronteiras (sintática, semântica e pragmática) são identificados juntamente com os processos de compartilhamento de conhecimento que eles demandam (transferência, tradução e transformação). No lado direito da Figura 5.2 observam-se também as características de cada processo de compartilhamento de conhecimento. É importante destacar que tais características são cumulativas, ou seja, em uma fronteira semântica é necessário que a característica da fronteira sintática também seja atendida e em uma fronteira pragmática é preciso que as características das fronteiras sintática e semântica também sejam atendidas.

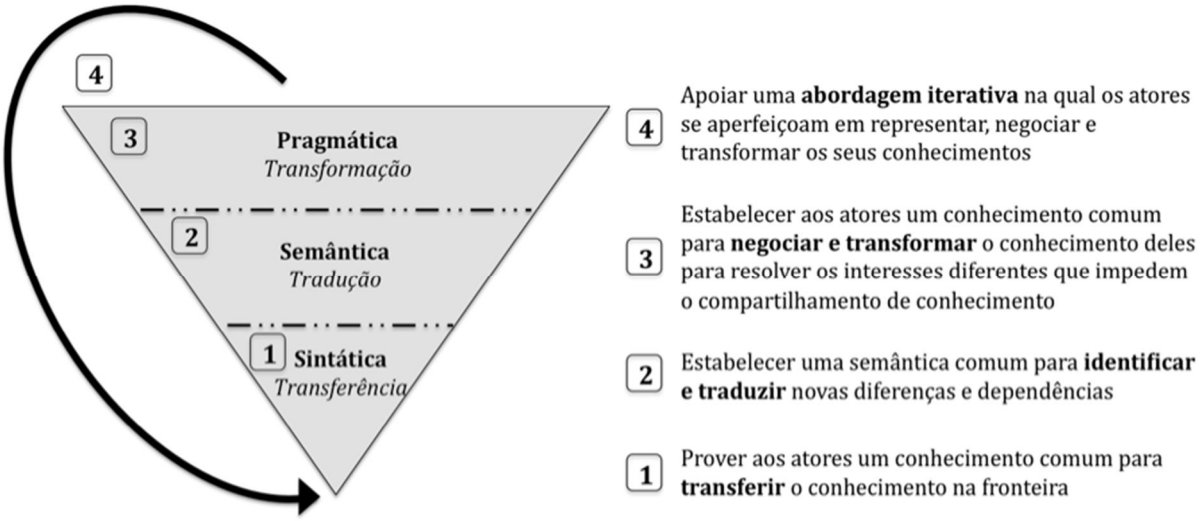

Figure 5.2 *Framework* de compartilhamento de conhecimento (Carlile, 2003)

A fronteira sintática é a mais simples. Nela as diferenças e dependências entre os atores são conhecidas e os conhecimentos não são tão especializados a ponto de gerar incerteza. Dessa forma, os atores podem lidar com os relacionamentos interfronteiras apenas usando uma sintaxe comum entre o ñremetenteò e o ñreceptorò da informação. O processo de compartilhamento de conhecimento requerido consiste em simplesmente *transferir* os diferentes conhecimentos para os diferentes domínios da fronteira. Por exemplo, ñsalárioò pode receber diferentes denominações: vencimento, remuneração, dividendo, etc. Criar uma sintaxe comum entre os diferentes domínios envolvidos seria correspondente a definir qual termo seria utilizado para representar ñsalárioò ao longo de todo o projeto. Com isso, basta fazer a *transferência* de conhecimentos entre os domínios que eles serão compreendidos, pois uma linguagem (sintaxe) comum foi adotada.

Conforme os conhecimentos dos diferentes domínios tornam-se mais especializados a abordagem sintática de transferência de conhecimento deixa de ser suficiente. Surge assim a fronteira semântica, na qual domínios diferentes geram diferenças interpretativas criando discrepâncias de significado. Por exemplo, um gerente faz uma estimativa de lucro para um mês com base no valor gasto com matéria prima, os salários mensais dos trabalhadores, o preço de venda do produto e a quantidade vendida. Entretanto, ele desconhece o fato de que o departamento financeiro paga o décimo terceiro salário dos funcionários em parcelas, sendo que um destes pagamentos ocorrerá justamente nesse mês. Com isso, se o gerente planejar um aumento na produção confiando nos lucros que calculou, a empresa pode acabar com saldo negativo, pois a parcela do décimo terceiro não foi considerada. Neste exemplo, o gerente e o departamento financeiro estão em domínios diferentes que precisam compartilhar informações de uma forma mais complexa. O processo requerido para esse compartilhamento passa a ser a *tradução* do conhecimento (Carlile, 2003). Neste processo, é necessário estabelecer uma semântica comum para identificar e *traduzir* as diferenças interpretativas e dependências entre as informações dos diferentes domínios. Para isso Carlile (2003) indica o uso de práticas compartilhadas que proveem uma base onde, através de aprendizagem e *tradução* de conhecimento, é possível criar ñnovos acordosò que ajudam a reconciliar as diferenças na fronteira. No caso do exemplo fictício citado, o uso de práticas compartilhadas evitaria o problema: se o gerente e o departamento financeiro realizassem essa projeção de lucro mensal de forma compartilhada, os conhecimentos ñtácitosò dos diferentes lados da fronteira seriam *traduzidos* para ambos os domínios de forma que as dependências entre esses domínios seriam melhor reconhecidas.

Quando os domínios de conhecimento são tão especializados que geram interesses diferentes, desenvolver um conhecimento comum adequado para ser compartilhado se transforma em um processo político e a fronteira passa a ser pragmática. O exemplo da Figura 5.1 se enquadra neste caso. Os domínios de conhecimento especializados da equipe do motor e da equipe de estilo criaram interesses diferentes quanto ao tamanho do motor: a primeira equipe criou um motor maior para responder aos seus requisitos de potência, enquanto a segunda equipe criou um compartimento menor para acomodar o motor para responder aos seus requisitos de *design*. Estes interesses além de diferentes são contraditórios. Nesse caso, é preciso considerar que o custo de um grupo receber o conhecimento do outro grupo não se resume ao custo de aprender o que é novo. É também necessário que os grupos ajustem ou *transformem* a sua forma corrente de realizar suas atividades para acomodar este

conhecimento proveniente do outro grupo e, assim, colaborar nesta fronteira. Voltando ao exemplo da empresa automobilística: os grupos de motor e estilo precisam negociar o conhecimento comum e ajustar o conhecimento de seus domínios para acomodar esse novo conhecimento e desenvolver um novo modelo do carro que atenda a requisitos de ambos. Em outras palavras, na fronteira pragmática cada grupo identifica as diferenças e dependências do conhecimento dos outros grupos que são relevantes para o seu próprio grupo, negocia alternativas na fronteira entre grupos e coletivamente transforma o conhecimento correntemente usado. Com isso, equipes, métodos de resolução de problemas e *boundary objects*[9] se tornam mais importantes (Carlile, 2003).

Por fim, é importante comentar que trabalhar com vários tipos de fronteiras traz consequências que não podem ser resolvidas em apenas uma ñetapaò, requerendo assim um processo iterativo de compartilhamento e avaliação de conhecimento, criação de novos acordos e realização de mudanças se necessário. Através desse processo iterativo os atores melhoram sua capacidade de representar e identificar as diferenças e dependências de conhecimento que sempre trazem consequências para a fronteira.

Este *framework* foi usado nesta pesquisa para compreender e interpretar os dados coletados e analisados nos capítulos anteriores, ajudando assim a dar um maior sentido eles, o que será discutido nas próximas seções.

5.3. Domínios de conhecimento na arquitetura de sistemas de TI

No capítulo anterior, na seção 4.2, um esquema genérico dos tipos de arquitetos foi definido na Figura 4.1 para representar os resultados encontrados nas empresas pesquisadas. Esta figura é repetida aqui por questões de praticidade como a Figura 5.3 a seguir.

[9] De acordo com Grinter (1999), *boundary objects* são objetos que são ao mesmo tempo plásticos o suficiente para se adaptar às necessidades e restrições locais das várias partes que o utilizam, e robustos o suficiente para manter a identidade comum em todas as localidades.

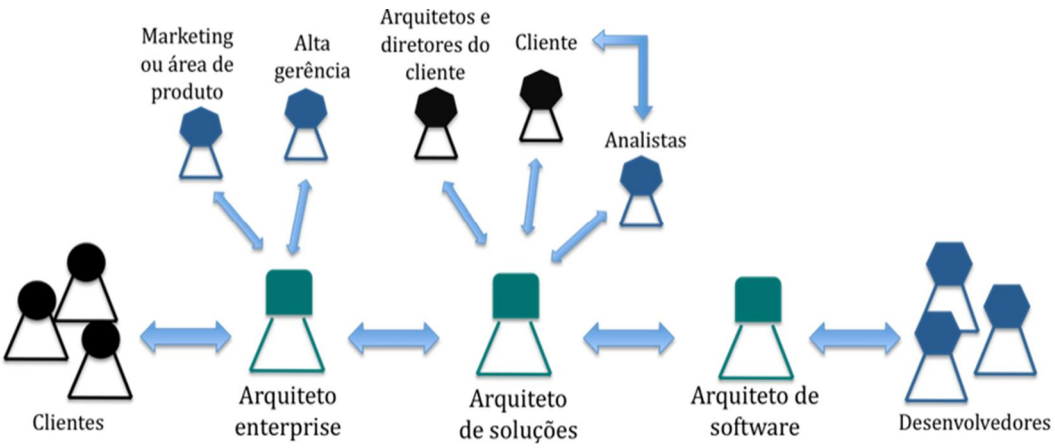

Figura 5.3 Estrutura genérica dos tipos de arquiteto e suas principais interações

Na seção 5.2 descreveu-se que domínios especializados de conhecimento geram fronteiras de conhecimento. Analisando a Figura 5.3 é possível identificar os principais domínios de conhecimento no desenvolvimento de sistemas: os clientes, os analistas, a gerência, a área de marketing e os desenvolvedores. Os clientes tem o conhecimento específico do seu negócio. Esse conhecimento pode ser extremamente específico no caso de o cliente ser de áreas como aviação, mineração, energia nuclear e outras. Por sua vez, os analistas tem a função de compreender o domínio do cliente. A ideia é que cliente e analistas ñfalem a mesma línguaò, ou seja, que estejam no mesmo domínio. Já a gerência possui conhecimento específico na área de negócios, planejamento, contratos, etc, configurando assim o seu próprio domínio. A área de marketing traz o conhecimento do mercado, das tendências e da concorrência. Por fim, os desenvolvedores possuem o conhecimento técnico de linguagens de programação, ferramentas, etc. É possível ainda agregar estes principais domínios de conhecimento em três esferas de conhecimento mais amplas, de acordo com o tipo de conhecimento especializado de cada domínio e com o foco das interações entre os atores destes diferentes domínios e das interações deles com os arquitetos (e consequente influência na arquitetura): organizacional, funcional e técnica. O grupo de marketing, a gerência, os clientes e arquitetos (*enterprise*) interagem com foco na definição do contrato e escopo, lidando com aspectos como custo, lucro, legados, *background* tecnológico do cliente e da empresa, entre outros, que influenciarão na arquitetura. *Stakeholders* do cliente também interagem com analistas e arquitetos (de soluções) com foco nas funcionalidades que o produto terá, e, através dessas interações, os requisitos serão identificados e estes constituem o principal insumo da arquitetura de software do projeto. Por fim, desenvolvedores interagem

com arquitetos (de software) com foco nos aspectos mais técnicos, ou seja, no desenvolvimento e codificação dos produtos em questão. Estes aspectos de tecnologia, desenvolvimento e codificação influenciam bastante na arquitetura de software, uma vez que ela precisa ser desenvolvida, e também na arquitetura de TI como um todo, pois a arquitetura de software deve estar de acordo com ela. Assim, os domínios de conhecimento do cliente (no que diz respeito à definição do contrato), da área de marketing ou de produto e da alta gerência são reunidos na esfera organizacional; os domínios de conhecimento dos analistas, de *stakeholders* do cliente e dos próprios clientes (no que diz respeito às características do produto que será desenvolvido) são reunidos na esfera funcional; e, por fim, o domínio de conhecimento dos desenvolvedores está localizado na esfera técnica. Essas esferas de conhecimento podem ser identificadas na Figura 5.4 a seguir, adapta a partir da Figura 5.3.

Essas esferas de conhecimento da Figura 5.4 compreendem assim os ñgrandesò domínios especializados de conhecimento da arquitetura de sistemas de TI e, portanto, permitem identificar as principais *fronteiras* de conhecimento nos projetos. As seções seguintes descrevem a influência das atividades dos diferentes tipos de arquiteto nesses domínios e em seguida analisam a classificação das empresas (descrita na seção 4.2 e subseções) sob esse ponto de vista (esferas de conhecimento x tipos de arquitetos). É importante ressaltar que estes não são os únicos domínios, nem consequentemente as únicas fronteiras do desenvolvimento de sistemas de software, mas foram os principais identificados nesta pesquisa.

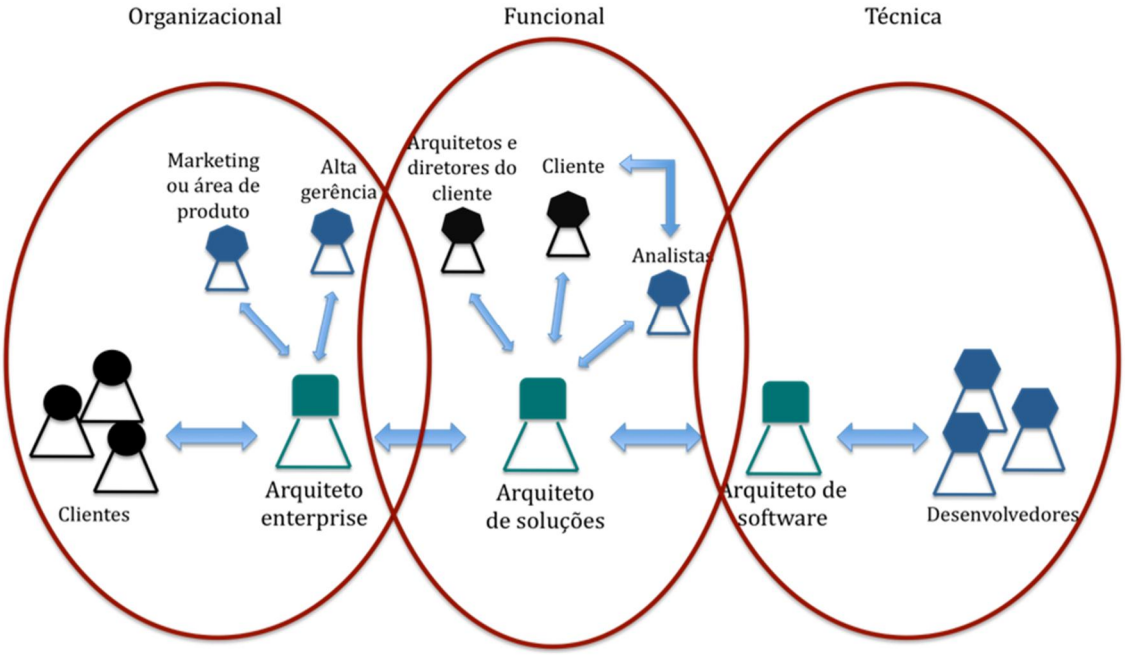

Figura 5.4 Representação das esferas de conhecimento

5.3.1. Arquitetos e as fronteiras de conhecimento

Analisando a Figura 5.4 percebe-se que cada arquiteto está em uma esfera diferente de conhecimento: o arquiteto *enterprise* está na esfera organizacional, o arquiteto de software está na esfera técnica e o arquiteto de soluções está numa esfera intermediária, chamada de funcional por estar diretamente ligada com a definição do produto que será produzido. Dessa forma, essa divisão de atividades também divide as fronteiras entre os arquitetos. Assim, ao invés de um único papel ter que lidar com todas as fronteiras, este esforço é dividido e o compartilhamento de conhecimento ocorre em ñetapasò ou grupos. Cada arquiteto agrega e difunde o conhecimento do e para o seu grupo específico. Os arquitetos *enterprise*, que estão bastante envolvidos com atividades gerenciais, contribuirão mais facilmente com o compartilhamento de conhecimento dos e para os domínios da gerência, do *marketing*, e dos clientes no que diz respeito ao contrato e definição do produto. Os arquitetos de soluções trabalham diretamente com a criação do produto (vivendo muitas vezes dentro do cliente) e, dessa forma, contribuirão mais facilmente com o compartilhamento de conhecimento dos e para os domínios dos clientes (do ponto de vista da solução e funcionalidades do produto) e dos analistas. Por fim, os arquitetos de software normalmente já foram programadores (muitos deles são desenvolvedores *sênior*), por isso compartilharão mais facilmente o conhecimento dos e para os desenvolvedores.

Assim, o fato de os diferentes tipos arquitetos atuarem em domínios diferentes contribui para o compartilhamento de conhecimento através das fronteiras, que podem ser sintáticas, semânticas e pragmáticas. Se a fronteira for do tipo sintática, os três tipos de arquitetos contribuem para a definição de uma sintaxe comum entre o ñremetenteò e o ñreceptorò do conhecimento (ver seção 5.2) que são de diferentes domínios, de forma que tal sintaxe considere o conhecimento das três esferas de conhecimento. O Informante 7 exemplifica a criação dessa sintaxe comum:

> ñ*Nós temos aqui um domínio de cobrança... no domínio de cobrança o que acontece? Pra mesma entidade física nós temos 4 ou 5 termos do vocabulário normal da organização. A gente não tem como trabalhar isso, porque o que acontece? Imagina só, 5 termos pra uma dívida, uma entidade dívida. Uma entidade dívida pode ser chamada de dívida, de contrato, de recebível, de obrigação. Mas calma aí, quem é que usa isso daí? Ah não, na parte da tecnologia é contrato. Na parte de negócio o que é? Ah, eles chamam de dívida mesmo. Então o que acontece? Isso* [vocabulário comum] *é importantíssimo pro arquiteto gerenciar ambiguidades do ambiente.*ò ï Informante 17, Empresa D.

Esse trabalho de criar uma sintaxe comum que permita que o conhecimento seja transferido através das fronteiras deve ser algo que acontece com a influência de todos os tipos de *stakeholders*, não apenas os arquitetos, uma vez que todos podem contribuir e se beneficiar de tal sintaxe. O Informante 16 sugere que haja reuniões de alinhamento dos termos em alguma fase do projeto, como exemplificado no trecho de entrevista a seguir. O ideal é que essas reuniões sejam como as que ocorrem na Empresa E: multidisciplinares, pois assim ao mesmo tempo em que é possível capturar a sintaxe específica de cada grupo, estes adquirirão a sintaxe comum criada na reunião.

> ñ*É necessário que exista uma linguagem mais comum entre todos os participantes do projeto. Então, quer dizer: não é o desenvolvedor falando lá um* ó*tecniquês*ô*, não é o analista de sistemas que só sabe falar a língua do negócio e o arquiteto ali no meio. É necessário que exista uma fase no projeto onde se defina os termos comuns, por exemplo: ah, o desenvolvedor vai estar lá falando* ó*o shipping etc, ect...*ô*, aí o analista de sistema vai estar falando* ó*A remessa...*ô*. É a mesma coisa, só que um está falando lá o componente de*

óshippingô que ele implementou, que já tem no programa ou em uma biblioteca etc, e o analista de sistema olhou pro negócio e viu ôbah! realmente as entregas são feitas em múltiplas remessas e talô Então é necessário que existam reuniões de alinhamento de termosò ï Informante 16, Empresa C.

No caso de uma fronteira semântica, domínios diferentes geram diferenças interpretativas criando discrepâncias de significado, e assim começam a surgir dependências entre os domínios que não são bem definidas ou conhecidas (ver seção 5.2). Por exemplo, um analista de negócios interpreta os dados de acordo com seu domínio (negócio), enquanto um desenvolvedor foca em lógica de programação, padrões e outros detalhes de código, isto é o domínio técnico. Com isso, eles podem interpretar um mesmo requisito de formas diferentes, até mesmo conflitantes. Para evitar esse problema, o conhecimento precisa ser traduzido entre os domínios. Todos os informantes afirmaram que fazer a tradução das necessidades dos clientes para termos técnicos é uma das suas principais atribuições, como ilustrado nos seguintes trechos de entrevistas:

ñEu acho que uma das grandes funções do arquiteto, é realmente essa, é fazer a tradução das necessidades de negócios pro time técnicoò ï Informante 16, Empresa C.

ñO arquiteto de software, ele tem que falar duas linguagens, a primeira é a técnica e a segunda é o idioma da organização do clienteò ï Informante 17, Empresa E.

ñÉ assim, no meu ponto de vista eu acho que esse é um dos objetivos do nosso trabalho: fazer essa tradução dessa quantidade de coisas que acontecem e produzir pro time de desenvolvimento, já de uma forma mais técnica, tirar um pouco dessa linguagem naturalò ï Informante 1, Empresa A.

Além disso, aqui se devem considerar as três esferas de conhecimento. Dentro de cada esfera de conhecimento existem diferentes domínios. Por exemplo, na esfera organizacional existe o domínio dos gerentes, o domínio do *marketing* e o domínio do cliente. Como cada arquiteto está mais familiarizado com a sua esfera de conhecimento, eles podem contribuir para a semântica comum dentro delas mais facilmente, pois conhecem as possíveis

discrepâncias interpretativas que podem surgir. Assim, ao realizar o seu trabalho, passam para o próximo arquiteto um conhecimento que já selecionou e agregou as informações dos diferentes domínios de sua esfera. Por exemplo, na esfera organizacional encontram-se três principais domínios de conhecimento: da gerência, dos clientes e do marketing. Uma fronteira semântica pode surgir se, por exemplo, um cliente pedir um produto similar ao da concorrência. A gerência pode interpretar esse pedido da seguinte forma: o cliente quer concorrer com seu concorrente, então o produto deve chegar o mais rápido possível ao mercado, pois o produto do concorrente já está em circulação. E assim usará seu conhecimento para sugerir marcos e entregas mais curtos. Por sua vez o time de marketing vai trazer o conhecimento de mercado, identificando, entre outras coisas, o que o produto do concorrente tem que constitui uma vantagem competitiva que o produto que será produzido deve atender. Se tal vantagem, por exemplo, for um alto poder de customização por parte dos usuários e tal poder de customização seja mais trabalhoso de desenvolver e por isso exija um tempo de desenvolvimento maior, surgem as diferenças interpretativas, uma vez que a gerência interpretou que o tempo de desenvolvimento deveria ser mais curto. Nesse cenário, o arquiteto *enterprise* pode contribuir justamente na identificação dessa discrepância de significado e auxiliar na tradução de conhecimento entre os domínios da gerência, do marketing e do cliente, para que ao fim o conhecimento tenha uma semântica comum. Assim, o arquiteto *enterprise* passa um conhecimento para o arquiteto de soluções no qual o conhecimento dos domínios reunidos na esfera organizacional já foi agregado, e o mesmo acontece nas outras esferas. Isso está alinhado com a solução apontada por Carlile (2003) para esta fronteira: a presença de ñ*brokers*ò (*boundary spanners*) ou tradutores, pois os diferentes arquitetos representam esse papel para suas esferas de conhecimento. Isso também está alinhado ao, e estende o, trabalho de Grinter (1999), no qual ela afirma que os arquitetos de software devem agir como *boundary spanners* nos projetos, uma vez que eles têm contato com diversos tipos de *stakeholders*. O que se encontrou nesta pesquisa é que não somente os arquitetos de software devem agir dessa forma, mas todos os tipos de arquitetos de TI devem, e na prática o fazem. Por exemplo, os três últimos trechos de entrevista são de arquitetos equivalentes ao de soluções (Informantes 16 e 17) e de software (Informante 1).

Por fim, na fronteira pragmática, traduzir significados diferentes não é mais suficiente, tornando-se necessário negociar esses significados e interesses entre os atores. Um exemplo real de uma fronteira pragmática entre as esferas funcionais e técnicas foi narrado pelo Informante 2 e transcrito no trecho a seguir:

ñ*Eu vou te dizer uma situação que eu enfrentei que não sei se vai responder bem isso. Teve um projeto que eu entrei logo após a definição da arquitetura, eu fiz tunning da arquitetura, e tinha um time fora, time de desenvolvimento. Eles não gostavam muito da arquitetura, não achavam muito legal, mas a gente fez o tunning e a arquitetura começou a fluir tranquilamente. Ela era bem conceitual, não era baseada em frameworks de mercado, arquitetura pura posso dizer. Ela era aplicação de conceitos arquiteturais puros, e ficou boa. Eu não sei se o time de lá tinha essa visibilidade, mas uma vez eu fui para lá e esse time me chamou pra conversar e eles disseram: ósabe o sistema tal, nós estamos pensando em mudar para óSpringôô, um framework Java que provê uma série de coisas prontas, inclusive injeção de dependência, óo que tu achas?ô. Nós teríamos que abandonar completamente a outra arquitetura e adotar injeção de dependência. Aí eu perguntei: ómas por que vocês querem isso?ô. óNós queremos isso para poder manter mais o código. Queremos dar mais manutenção no código e escrever menos código de infraestruturaô. Porque a arquitetura anterior tinha mais código de infraestrutura para mantê-la. Era como se fosse um framework, tinha muito código pra manter. Uma vez que ela estava pronta e estabilizada eu só tinha o trabalho de inserir as regras de negócio, mas eu poderia eventualmente ter que dar manutenção no core da arquitetura. Já a decisão do time de escolher um framework pronto era para se livrar desse tipo de coisa e pegar algo já aprovado pelo mercado como sendo algo bem, escalável, etc, para facilitar a codificação. Então vocês veem: eram dois times em locais diferentes, que tinham visões diferentes. Um time queria fazer uma arquitetura própria, independente de mercado, e o outro queria utilizar um framework de mercado que já foi provado que é bom, e que funciona. E eles foram adiante, refizeram todo código e a nova arquitetura foi colocada no ar. Ambas funcionaramò* ï Informante 2, Empresa B.

É possível então concluir que existia uma fronteira pragmática entre as esferas funcional (onde o arquiteto que definiu a arquitetura inicial estava) e a técnica (onde os desenvolvedores estavam) e cada um dos domínios tinha um conhecimento bastante especializado que requeria negociação para que a solução atendesse ambos os lados, o que não aconteceu. Com isso, apenas um lado da fronteira estava satisfeito com a solução adotada, o arquiteto, e, no momento em que este arquiteto saiu do projeto, resolveu-se refazer toda a arquitetura. Ambas as soluções funcionavam, mas a primeira só atendia a um lado da

fronteira. Nesse cenário, é necessário ocorrer a *transformação* do conhecimento, pois os grupos precisam ajustar a sua forma atual de realizar atividades para acomodar o conhecimento proveniente dos outros grupos e, assim, colaborar de maneira efetiva. Neste ponto, as interconexões entre os arquitetos se tornam ainda mais importantes. A divisão de tipos de arquitetos é benéfica na fronteira semântica, uma vez que também divide as fronteiras com as quais eles precisam trabalhar. Entretanto, após essa divisão restam ainda três fronteiras principais derivadas dos diferentes domínios: organizacional, funcional e técnica. Geralmente, especialmente em grandes projetos, o conhecimento de cada domínio destas fronteiras é muito especializado, o que pode trazer consequências ruins para o outro lado da fronteira. Por exemplo, um conhecimento sobre uma legislação do domínio gerencial traz grande quantidade de novidade para o domínio funcional, uma vez que ela deverá ser considerada durante a criação do produto. As funcionalidades requisitadas pelo cliente para o domínio funcional impactam como novidade no domínio gerencial do ponto de vista de cronogramas e planejamentos, por exemplo. A definição de como a arquitetura será construída (domínio funcional) pode trazer muita incerteza para o domínio técnico dos desenvolvedores, caso, por exemplo, eles não estejam acostumados a trabalhar com o estilo arquitetural em questão. E o impacto inverso existe caso um desenvolvedor resolva não respeitar um aspecto da arquitetura porque este não é o melhor do ponto de vista técnico. Dessa forma, apesar de o trabalho de cada tipo de arquiteto ajudar no compartilhamento de conhecimento nas suas esferas, o conhecimento ainda precisa ser compartilhado e negociado *entre estas esferas*: é aí que entra a importância das interconexões e interdependências entre os tipos de arquitetos. Eles não podem trabalhar isolados, caso contrário o conhecimento comum tão desejado em projetos de desenvolvimento de software ficará prejudicado; algum domínio acabará não compartilhando corretamente seu conhecimento através da sua fronteira. Voltando ao exemplo do Informante 2 citado anteriormente, caso o arquiteto do projeto que estava na esfera funcional interagisse (com foco nos aspectos arquiteturais) com alguém da esfera técnica, a arquitetura poderia ser negociada para atender aos dois domínios.

Os diferentes tipos de fronteira aumentam o nível de dificuldade em compartilhar conhecimento. Tal nível de dificuldade está ligado ao nível de novidade e de dependência entre o conhecimento dos diferentes grupos. Pode ser que um projeto tenha um conhecimento ñsimplesò que gere apenas fronteiras sintáticas. E pode também ocorrer de um projeto possuir um conhecimento tão complexo que todas as suas fronteiras entre os diferentes domínios sejam semânticas ou pragmáticas. Assim, os processos necessários para compartilhar o

conhecimento variam de transferência a transformação (passando por tradução).Um dos principais pontos identificados sobre o trabalho dos arquitetos é que eles ajudam justamente nesses processos de compartilhamento de conhecimento de forma que as necessidades do cliente, em linguagem natural, sejam transferidas, traduzidas e transformadas até que alcancem os termos técnicos que podem ser desenvolvidos pelos desenvolvedores, criando com isso um conhecimento comum no projeto.

Em resumo, é possível concluir que os arquitetos são importantes para todos os tipos de fronteiras: o compartilhamento de conhecimento para o desenvolvimento de uma solução de software se dá através dos diferentes tipos de arquitetos tal que cada um deles seleciona e agrega informações, trabalha com elas e as passa adiante. Cada arquiteto desempenha uma função diferente neste ñprocessoò e cada um deles contribui de uma forma diferente para o compartilhamento de conhecimento.

Por fim, segundo o que se discutiu nesta seção, não apenas os arquitetos de TI, como também a divisão das atividades de arquitetura em tipos de arquitetos, contribuem para a superação destas fronteiras (seja qual for o tipo). Entretanto, conforme descrito na classificação das empresas da seção 4.2.1, as empresas não dividem as atividades de arquitetura da mesma forma. A seção seguinte discute a classificação das empresas considerando os diferentes domínios e fronteiras de conhecimento.

5.3.2. Classificação das empresas

Como definido no Capítulo 4 (seção 4.2 e subseções), as empresas foram classificadas em 3 grupos de acordo com a definição dos tipos de arquitetos. Relacionando essa classificação com as principais esferas de conhecimento percebe-se que no grupo chamado ñTipos de arquitetos definidosò as três esferas são bem diferenciadas e suas interconexões definidas, ou seja, há um tipo de arquiteto trabalhando em cada esfera e os três tipos interagem entre si. As empresas classificadas nesse grupo (Empresas A e E) são multinacionais, desenvolvem muitos projetos paralelos e trabalham com DDS, com isso especula-se que as fronteiras que surjam sejam principalmente do tipo pragmática, pois os conhecimentos de cada domínio passam a ser mais especializados. Esta fronteira é a que traz mais dificuldades para o compartilhamento de conhecimento, devido justamente a estes domínios bastante especializados e todas as negociações e ajustes necessários para resolver interesses diferentes relacionados a eles. Por isso, o papel dos arquitetos se torna mais importante e, como discutido na seção anterior, a própria divisão em tipos de arquitetos e as interconexões entre

eles podem contribuir bastante para a superação desse tipo de fronteira. Portanto, estas empresas se beneficiariam mais de uma configuração com os três tipos de arquitetos definidos, pois desta forma estes contribuiriam melhor para o compartilhamento de conhecimento.

Por outro lado, as empresas alocadas na categoria ñTipos de arquitetos parcialmente definidosò não apresentam o tipo de arquiteto *enterprise*, mas o arquiteto de soluções já realiza algumas de suas atividades. Dessa forma, as atividades de arquitetura da esfera organizacional começam a aparecer, mas ainda são realizadas pelo arquiteto de soluções, que está na esfera funcional. As empresas desta classificação (Empresas C, D e I) formam um grupo intermediário entre as empresas pesquisadas e, por isso, especula-se que esta configuração intermediária seja suficiente para elas no contexto em que elas se encontravam durante a pesquisa. Entretanto, caso essas empresas cresçam e se assemelhem às do primeiro grupo, talvez seja necessário separar melhor as atividades de arquitetura da esfera organizacional para favorecer mais o compartilhamento de conhecimento.

Por fim, as empresas classificadas no grupo ñTipos de arquitetos não definidosò (Empresas F e G) apresentam atividades de arquitetura apenas nas esferas funcional e técnica, e mesmo estas não apresentam uma separação tão clara. Isso acontece porque nas empresas reunidas nessa classificação os arquitetos não participam das reuniões de definição de escopo e contrato e nem tem contato com os clientes. Dessa forma, não há um ñrepresentante de arquiteturaò na esfera de conhecimento organizacional.

Essas empresas desenvolvem soluções para apenas um único cliente, possuindo um projeto maior que engloba os projetos menores que os clientes requisitam; elas são totalmente colocalizadas e possuem uma equipe de quatro arquitetos que não tem atividades diferenciadas de forma institucionalizada. Com isso, os próprios arquitetos se encarregam de fazer essa divisão. No cenário destas empresas, especula-se que as fronteiras pragmáticas sejam menos frequentes. Sendo assim, uma divisão formalizada de tipos de arquitetos pode não ser tão importante, pois os arquitetos podem conseguir lidar com as fronteiras usando a divisão que eles mesmos realizam em cada projeto. Entretanto, se porventura essas empresas crescerem mais a ponto de suas fronteiras se tornarem mais complexas (como ocorre, por exemplo, nas Empresas A e E) pode ser necessário estruturar esse time de arquitetos de uma maneira mais formalizada para atender os diferentes domínios especializados de conhecimento que se formam.

A Figura 5.5 a seguir ilustra a presença das esferas de conhecimento em cada uma das classificações. A esfera organizacional é representada sobreposta à esfera funcional na classificação ñTipos de arquitetos parcialmente definidosò porque um mesmo ator realiza as atividades de arquitetura de ambas as esferas, então é como se as atividades de arquitetura da esfera organizacional estivessem começando a surgir a partir das atividades da esfera funcional. Por outro lado, a esfera organizacional não é representada na classificação ñTipos de arquitetos não definidosò porque não há um ator que desenvolva as atividades de arquitetura nessa esfera.

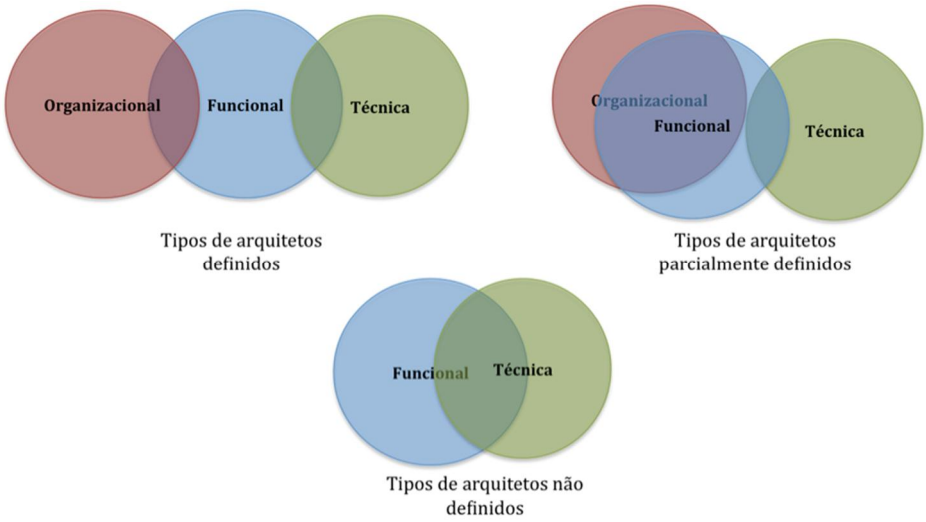

Figura 5.5 Esferas de conhecimento e a classificação das empresas

Em resumo, os resultados deste estudo sugerem que conforme as empresas crescem, desenvolvem mais projetos paralelos e usam desenvolvimento distribuído de software mais frequentemente, as especializações dos arquitetos se tornam melhor definidas e mais necessárias. Isso pode ser relacionado com os mecanismos de tradução e transformação de conhecimento, uma vez que, nesse cenário, os conhecimentos dos domínios são mais especializados e as diferenças e dependências desses conhecimentos são ressaltadas, aumentando assim o esforço requerido para compartilhar o conhecimento. Por exemplo, as empresas no grupo ñtipos não definidosò trabalham com desenvolvimento colocalizado. Neste cenário, todos os grupos de *stakeholders* estão localizados no mesmo espaço físico (exceto os clientes), com isso a tradução e a transformação são facilitadas, uma vez que dúvidas podem ser facilmente resolvidas com o ator que possui este conhecimento. Nas empresas classificadas no grupo ñtipos definidosò, isso não é tão simples, pois este ator pode estar em outro país, falando um idioma diferente e trabalhando em um fuso horário diferente.

É importante ressaltar que as empresas precisam adotar uma configuração que seja adequada a sua realidade, considerando sua estrutura, seus clientes, sua equipe e outros aspectos bastante particulares. O que este trabalho identificou é que, nas empresas pesquisadas, normalmente há uma divisão pelo menos entre as atividades de arquitetura de solução e de software. Dessa forma, os resultados desta pesquisa sugerem que tal divisão é útil para uma empresa que esteja classificada em qualquer um dos três grupos. Além disso, isto é um indicativo de que a academia não pode continuar estudando a arquitetura de software de forma isolada e poderia, pelo menos, considerar esta divisão de atividades, além de também abordar as atividades mais organizacionais. E, por fim, é preciso que estes tipos de arquitetos tenham uma boa interação entre si e que estas interações sejam apoiadas através de ferramentas, pois suas interconexões são muito úteis para um bom compartilhamento de conhecimento.

5.3.3. Hipóteses e oportunidades de pesquisa

Essa discussão sobre a participação dos diferentes tipos de arquitetos no compartilhamento de conhecimento e criação de conhecimento comum dentro dos projetos acaba gerando hipóteses e oportunidades de pesquisa. Tais hipóteses estão relacionadas a o que acontece quando as condições nas quais os projetos são desenvolvidos variam e podem ser exploradas em trabalhos futuros. Algumas destas hipóteses serão discutidas nessa seção.

5.3.3.1. A distribuição geográfica dos arquitetos

Como mencionado na seção anterior sobre a classificação das empresas (5.3.2), os dados sugerem que conforme as empresas crescem, desenvolvem mais projetos paralelos e utilizam mais DDS elas poderiam se favorecer mais de uma maior divisão e especialização das atividades de arquitetura e a presença de diferentes tipos de arquitetos. Tal divisão pode ser útil em projetos distribuídos, pois os arquitetos podem estar localizados em diferentes localidades (como na Empresa A) e facilitar o compartilhamento de conhecimento entre os domínios dessas diferentes localidades. Entretanto, como se dá a interação, as interconexões e o compartilhamento de conhecimento entre esses diferentes tipos de arquiteto é algo que não foi explorado nesta pesquisa e representa uma boa oportunidade de pesquisa. Por exemplo, na Empresa A o arquiteto *enterprise* está sempre localizado na matriz da empresa nos Estados Unidos, o arquiteto de soluções está na maioria das vezes dentro do cliente da empresa, e cada uma das filiais que participam do desenvolvimento possui um arquiteto de software. Segundo

o Informante 1, esses diferentes tipos de arquitetos interagem principalmente através de reuniões no decorrer do projeto. Como essas reuniões ocorrem, como as informações são passadas de uma localidade para a outra e de um arquiteto para o outro, quais as dificuldades relacionadas à distância e quais atividades os arquitetos realizam para superá-las são pontos interessantes de serem investigados posteriormente.

Tal investigação pode ser feita através de um novo estudo qualitativo que entreviste os diferentes tipos de arquiteto de cada localidade da empresa, com a realização observações não participativas das reuniões entre eles ou com acompanhamento de diferentes fases de um projeto que possua essas características.

5.3.3.2. O papel do Arquiteto de negócios

Outro ponto interessante de ser verificado é a questão do arquiteto de negócios. Como mencionado anteriormente, este tipo de arquiteto sugerido pela IASA (Akenine, 2008) não foi encontrado em nenhuma das empresas pesquisadas. Assim como o arquiteto *enterprise*, este tipo de arquiteto está mais relacionado a aspectos estratégicos da empresa. Com isso, especula-se que a presença bem definida deste tipo de arquiteto seja mais interessante para empresas grandes, como as do grupo chamado de ñTipos de arquitetos definidosò, (o que também é especulado para o arquiteto *enterprise*). Entretanto, dentre as empresas pesquisadas, apenas a Empresa A e a Empresa E estão neste grupo e em nenhuma das duas foi possível visitar e entrevistar algum arquiteto que trabalhasse nas matrizes das empresas (localizadas nos Estados Unidos e Espanha respectivamente) e, como o arquiteto de negócios seria um papel mais estratégico é mais provável que ele estivesse localizado nas matrizes. Além disso, as atividades deste tipo de arquiteto também podem ser realizadas por outros papéis, localizados nas matrizes, que não são chamados de arquitetos (como foi encontrado para os outros tipos em várias das empresas pesquisadas), o que poderia explicar o fato de os arquitetos entrevistados não mencionarem um arquiteto de negócios ou similar.

Mais uma vez: tal aspecto pode ser explorado através da extensão da pesquisa para as matrizes das empresas.

5.3.3.3. Tipos de projetos

Outro aspecto interessante de se aprofundar está relacionado com o que ocorre com a divisão das atividades de arquitetura de acordo com o tipo de projeto. Por exemplo, a maioria das empresas pesquisadas produziam sistemas para um cliente ou vários clientes, sendo que estes

compram esses sistemas e passam a ser ñdonosò deles (respeitando o tipo de licença de cada um). Nesse cenário, foi observado que a ñlocalizaçãoò dos diferentes tipos de arquitetos pode ser interessante. O arquiteto *enterprise* é o que está mais próximo ao mercado e ajuda a identificar as necessidades e possíveis novos clientes. O arquiteto de soluções pode trabalhar dentro da própria empresa cliente, fazendo uso dessa proximidade para compreender melhor as necessidades do cliente e contribuir para a criação de um produto que atenda essas necessidades. Por fim, o arquiteto de software trabalha próximo aos desenvolvedores, logo dentro da empresa que desenvolve o produto. Esse foco diferenciado dos diferentes tipos de arquitetos contribui para que aspectos importantes de diferentes esferas de conhecimento (organizacional, funcional e técnica) sejam considerados no produto que será entregue para a empresa cliente.

Entretanto, existe o conceito de software como serviço (Carraro e Chong, 2007). Neste caso, a empresa não produz um produto para um cliente, ela cria um produto que será oferecido como serviço para vários clientes. Os clientes não serão donos do produto, eles pagarão pelo serviço que aquele produto oferece. Com isso, a configuração dos tipos de arquitetos encontrados nesta pesquisa provavelmente seria diferente neste contexto de software como serviço. Por exemplo, esse produto não tem um único cliente como foco, logo não há como ter um arquiteto (nesse caso o de soluções) dentro deste cliente para compreender suas necessidades. Com isso, pode ser que os papéis de arquiteto de soluções e de software sejam unidos em um mesmo papel. Em resumo, torna-se interessante investigar como os diferentes tipos de projetos de software influenciam na organização das atividades de arquitetura. O tamanho da empresa, sua maturidade, seu negócio e seu tipo de desenvolvimento são fatores que influenciam na necessidade de diferentes tipos de arquitetos (tais aspectos foram observados neste trabalho), mas além deles o tipo de projeto também pode alterar a configuração encontrada, uma vez que a melhor estratégia sempre será a adaptação de acordo com a realidade da empresa e do projeto. Portanto, investigar mais esse aspecto que influencia na organização das atividades de arquitetura pode contribuir como mais um exemplo de diferentes abordagens que podem ser adaptadas de acordo com a realidade das empresas.

5.3.3.4. Métodos ágeis

Atualmente, percebe-se o crescimento do debate e uso de métodos ágeis (Dybå e Dingsøyr, 2008)(Goldman e Katayama, 2010)(Corbucci et al, 2011). Diferentemente das abordagens

mais tradicionais, estes métodos focam em criar produtos funcionais em releases que ocorrem mais cedo e com mais frequência e usam técnicas mais colaborativas para isso, como *pair programming*, refatoração e a presença de clientes trabalhando em conjunto com a equipe de desenvolvimento (Reifer, 2002). O Manifesto para o Desenvolvimento Ágil de Software (Agile Manifesto, 2001) estabelece um *framework* comum para processos ágeis de forma a valorizar os seguintes aspectos:

- Indivíduos e interações mais que processos e ferramentas;
- Software funcionando mais que documentação compreensiva;
- Colaboração do cliente mais que negociação do contrato; e
- Responder à mudança mais que seguir um plano.

Com isso, geralmente em métodos ágeis não á necessariamente uma divisão funcional de papéis tradicionais, prevalecendo a existência de equipes multifuncionais, onde uma pessoa pode representar papéis diferentes como analista, desenvolvedor, testador, etc, dependendo do contexto. Além disso, as equipes são auto-gerenciáveis e desenvolvem de forma iterativa, projetando, codificando, testando e documentando cada ñrequisitoò desenvolvido pela equipe. Dessa forma, é possível que a divisão de papéis de arquitetura, aqui chamados de tipos de arquitetos, seja alterada. Na verdade especula-se que duas coisas podem acontecer: a) a diferenciação de tipos de arquitetos pode não ser tão notável quanto no estudo aqui descrito, ou b) os tipos de arquitetos sejam representados por diferentes papéis (e diferentes em cada iteração ou Sprint). Durante a pesquisa não foi possível aprofundar nesse aspecto, mas este é um ponto interessante de se pesquisar e verificar como as diferentes atividades de arquitetura identificadas neste trabalho são realizadas no âmbito dos métodos ágeis.

5.3.3.5. Rastreabilidade de software

Por fim, outra hipótese que constitui um interessante assunto a ser pesquisado se refere a como esses tipos de arquitetos podem contribuir para a rastreabilidade de software (Spanoudakis e Zisman, 2005). A configuração de tipos de arquitetos encontrada neste trabalho situa ao menos um arquiteto em diferentes etapas do desenvolvimento de software. Considerando que os arquitetos constituem a principal fonte de informações dentro dos projetos (conforme identificado por Unphon e Dittrich (2010) e neste trabalho), especula-se que eles possam contribuir para a rastreabilidade de software. Ainda nesse sentido, destaca-se a importância da comunicação entre os diferentes tipos de arquitetos, pois desta forma os arquitetos repassam as informações das diferentes esferas de conhecimento onde atuam e

essas informações são muito importantes para a rastreabilidade, pois permitem que dados de uma esfera sejam relacionados com o que é desenvolvido na outra. Isso é ainda mais ressaltado pelo fato de a documentação não ser tão rigorosa em muitos projetos. Dessa forma, um interessante ponto de pesquisa é não apenas verificar como as informações são transmitidas, mas também associar essa transmissão com a rastreabilidade e os documentos que são produzidos. Através disso pode ser possível sugerir algum tipo de documentação ou ferramenta que favoreça a documentação dessa troca de informações, uma vez que ela pode ser importante para a rastreabilidade e a documentação do *rationale* nos projetos.

5.3.4. Requisitos

No capítulo anterior alguns aspectos relacionados à engenharia de requisitos foram descritos com base no que os informantes relataram nas entrevistas. A maior parte destes aspectos ou enfatizava a importância dos requisitos para o trabalho dos arquitetos ou tratava de problemas relacionados a eles. Os problemas citados pelos informantes estão principalmente relacionados com um problema clássico de engenharia de requisitos comumente representado pela Figura 4.9 da seção 4.5.

Sistemas de software são construídos para atender uma necessidade do cliente, seja ele externo ou interno, ou do mercado (o que visa atrair clientes). Dessa forma, os requisitos são parte fundamental do desenvolvimento desses sistemas e devem nortear todo o processo de desenvolvimento. Entretanto, todos os informantes afirmaram que enfrentam problemas com requisitos, especialmente o problema de desenvolver corretamente o que foi especificado e ao final o que foi desenvolvido não representar o que o cliente queria. Como mencionado anteriormente, este é um problema clássico da engenharia de requisitos e pode ocorrer por vários motivos. Um destes motivos está relacionado com a presença dos arquitetos em todas as fases do projeto. Nas Empresas F e G, por exemplo, os arquitetos geralmente não participam das reuniões iniciais de definição dos requisitos do produto: eles já recebem a lista de requisitos pronta dos analistas. Seria interessante que os arquitetos trabalhassem mais próximos aos analistas e também tivessem maior contato com os clientes, pois os arquitetos podem ajudar a capturar requisitos (especialmente os não funcionais) através de sua experiência técnica e organizacional que pode ser somada a experiência dos analistas e assim auxiliar na captura mais completa dos requisitos. Isso é interessante uma vez que os clientes descrevem as funcionalidades do produto que desejam, mas geralmente é necessário interpretar essas descrições para identificar os requisitos não funcionais do sistema. Esse é um

dos fatos pelos quais é recomendado que os arquitetos façam parte de todo o processo de desenvolvimento (Hofstader, 2008)(Taylor *et al.*, 2010), pois assim eles podem contribuir para evitar esse tipo de problema.

Com a divisão em tipos de arquitetos esse aspecto também pode ser favorecido, pois, como dito antes, a divisão em três tipos de arquitetos diferentes acaba criando três esferas de conhecimento: organizacional, funcional e técnica. E em cada uma dessas esferas haverá ao menos um arquiteto. Assim, o arquiteto *enterprise* poderá contribuir para a elicitação dos requisitos organizacionais ou gerenciais, como regulamentações, padrões da empresa ou aspectos financeiros por exemplo. O arquiteto de soluções poderia contribuir com os requisitos funcionais e não funcionais (especialmente estes últimos) do cliente, com as funcionalidades e qualidades do produto em si. E por fim o arquiteto de software contribuiria para os requisitos técnicos provenientes dos desenvolvedores, relacionados com a viabilidade técnica de se desenvolver (ou não) a solução. Todos esses ñtiposò de requisitos são importantes e se complementam para, juntos, formarem o produto que o cliente requisitou.

No caso das Empresas F e G, não há um arquiteto envolvido na esfera organizacional e mesmo o envolvimento na esfera funcional é bastante reduzido, uma vez que eles não participam das reuniões de definição de contrato (ambas), não interagem com clientes (principalmente a Empresa F, uma vez que a G ainda interage mesmo que muito pouco) e mesmo a interação com analistas não é muito frequente (na Empresa F principalmente). Como os analistas nesses casos concentram a interação com os clientes duas coisas podem acontecer: (i) o analista não capturar o requisito corretamente e (ii) o arquiteto não compreender corretamente o que o analista lhe passou. Em ambos os casos essas empresas poderiam se beneficiar se ao menos um arquiteto tivesse um contato maior com o cliente e trabalhasse junto com analistas na identificação e discussão dos requisitos.

O mesmo problema também pode estar relacionado com falhas no compartilhamento de conhecimento. Se a definição do requisito que corresponde a uma necessidade do cliente falhar em algum momento, o desenvolvimento realizado a partir dela irá cascatear o erro. Assim como um problema poderia ter ocorrido no início do processo, quando os requisitos são elicitados por analistas e arquitetos, um problema também poderia ter ocorrido durante o processo, o que poderia estar relacionado com a transferência, tradução e transformação de conhecimento descritas no *framework* de Carlile (2003). Com isso, uma possível forma de ajudar a reduzir esse problema também poderia ser dar mais atenção a esses processos e, de acordo com o que foi explicado na seção anterior, nas interconexões entre os tipos de

arquitetos e no compartilhamento de conhecimento entre as três principais esferas de conhecimento das quais os principais tipos de arquiteto fazem parte.

Os informantes também reclamaram sobre revisões de requisitos que não são feitas pelos *stakeholders* (problema descrito na seção 4.5). De fato, revisões também constituem uma forma de evitar o problema de se desenvolver um requisito que não representa a necessidade do cliente, não importando onde o erro ocorreu. A ideia é que, com as revisões, seja possível identificar problemas no rastreamento dos requisitos desde a elicitação até o desenvolvimento. Essas revisões não precisam nem ser revisões formais, podem constituir apenas a leitura de um documento antes da reunião onde ele será discutido. Entretanto, o que os informantes relataram é que os *stakeholders* consideram as revisões tarefas tediosas e eles não possuem uma ferramenta que motive ou demonstre a importância dessa atividade (seção 4.5). Além da falta de uma motivação maior para fazer revisões, estas também são prejudicadas por maus hábitos de *stakeholders* ao criarem seus artefatos (sejam eles formais ou informais) e requisitarem uma revisão a algum colega, como por exemplo realizar mudanças sem informar em que parte de um documento elas estão. Esse aspecto de hábitos é exemplificado pelos informantes da Empresa A:

> *Informante 15:* [o stakeholder] *manda todos os recursos: está aqui o código, está aqui o documento que se refere aquele código e eu inclusive já liguei o track changes do Word e já salientei pra ti as partes do documento que alteraram desde a última vez que tu revisaste, sabe? São esses pequenos detalhes assim que... esse é o cara que eu gosto de ajudar.*
>
> *Informante 14: Em contrapartida tem aquele outro que tem um documento de duzentas páginas, o cara fez uma alteração nele e manda o documento todo pra ti revisar a alteração dele. Tu não fazes a menor ideia de onde aconteceu, claro.*
>
> *Informante 1: O cara escreveu um parágrafo a mais e te mandou o documento sem track changes ligado.*
>
> *Informante 14: Então, aí obriga o revisor a fazer um diff do documento pra tentar achar onde está e aí começa problema. E as pessoas dizem assim "bah não deu tempo de revisar". Quando chegar no dia ele vai dizer: olha me mostra aí o que foi que tu fizeste.*

Para tentar melhorar este problema de revisões, é necessário incentivar mais os *stakeholders* a realizarem esta tarefa, utilizar ferramentas, pois estas podem deixar a atividade menos tediosa, e incentivar bons hábitos que facilitem a tarefa dos revisores para assim, também, tentar tornar a tarefa menos tediosa.

Por fim, esse problema de requisitos que são desenvolvidos sem representar a necessidade do cliente pode ser originado do próprio cliente, pois este pode ou não ter certeza de todos os aspectos do seu produto ou mudar de ideia muito constantemente. Entretanto, mesmo assim é responsabilidade da equipe (como um todo) que desenvolverá o sistema capturar essas necessidades. Este fator pode ser diminuído com a utilização de protótipos, pois, com estes, os clientes podem ver como o produto está sendo desenvolvido em várias etapas e isso o ajuda a refinar a sua própria ideia do que é o produto. Além disso, as empresas também devem deixar claro para os clientes o custo do retrabalho, de preferência de forma quantitativa (custo financeiro) até mesmo como uma garantia para a própria empresa de não ter mais trabalho do que o acordado devido a indefinições do cliente.

5.3.5. Documentação

A seção 4.4.1 descreveu alguns aspectos identificados nas empresas relativos à documentação que elas produzem. Dentre esses aspectos dois se destacam: a adaptação da documentação e os arquitetos como fonte de informação.

O primeiro aspecto diz respeito à quantidade e formalidade da documentação produzida. O que se notou foi que a maioria das empresas não dispende tanto tempo e esforço para produzir documentações extensas e muito completas: elas produzem apenas o necessário. Entretanto, este ñnecessárioò varia de acordo com o projeto e com a equipe que nele trabalha. Os documentos relativos à arquitetura (geralmente de software) serão usados como exemplo. Na Empresa E, o documento de arquitetura de software é um artefato simples e em alto nível que segue um padrão nos projetos. É constituído de um fluxograma que representa os componentes do sistema em caixas macro. Algumas dessas caixas são posteriormente descritas em um novo fluxograma do mesmo estilo até alcançarem um nível que represente a regra de negócio (ver seção 4.4.1). Além disso, se porventura algum componente precisar de alguma descrição extra, eles criam um documento com ela e colocam tudo (documento e fluxograma) em uma wiki. O quanto essas caixas macro serão refinadas e o quanto de descrição extra será necessário é algo que vai depender do projeto em questão. Na Empresa F a documentação é na sua maioria concentrada no Javadoc, ou seja, em código (seção 4.4.1).

Por outro lado, na Empresa A, os informantes afirmaram que o nível de detalhes da documentação depende do nível de maturidade da equipe. Quanto mais madura a equipe, menos específico o arquiteto precisa ser nas documentações. Na Empresa D, o informante 17 também relatou que a documentação depende do projeto, mas no geral o documento não é algo rígido e o arquiteto o desenvolve da maneira que achar apropriada (seção 4.4.1). A Empresa C por sua vez possui um documento de arquitetura mais completo, porém um dos informantes afirmou que normalmente eles também usam o código para fazer uma engenharia reversa e gerar a documentação; que o processo tradicional de projetar primeiro e depois gerar código é usado apenas para módulos e componentes críticos (seção 4.4.1).

Com isso, o que se observa é que existe documentação, mas é uma documentação menos extensa do que a academia recomenda e, apesar disso, isso funciona para as empresas. A explicação para isto é que a maioria das informações é passada através de reuniões. Em todas as empresas verificam-se reuniões frequentes, especialmente entre os tipos de arquitetos similares aos arquitetos de soluções e de software. E mesmo entre os três tipos de arquitetos identificados nesta pesquisa o que se verifica é que não há um documento padronizado para a troca de informações entre eles: isso também ocorre através de reuniões. Com isso, é através dessas reuniões (com o auxílio de alguns artefatos de alto nível) que os arquitetos normalmente passam as informações para suas equipes, estejam eles em qualquer uma das esferas de conhecimento citadas.

Este fato está relacionado com o outro aspecto citado no início desta seção: o arquiteto como fonte de informação. Com a análise dos dados das entrevistas, notou-se que na prática os arquitetos são a fonte de informação dentro das empresas. Mesmo quando as empresas constroem documentos mais completos, como a academia sugere, ainda assim os *stakeholders* preferem recorrer aos arquitetos em caso de dúvidas. Talvez por isso as empresas estejam diminuindo a ênfase em documentação, uma vez que seria dispendido um trabalho significativo para algo que não seria usado com tanta frequência. Este resultado está alinhado com o trabalho de Unphon e Dittrich (2010), no qual elas identificaram que não existe um grande esforço na documentação da arquitetura de software nas empresas onde elas pesquisaram e que esta existe implicitamente em discussões no decorrer do desenvolvimento. Elas relataram que o arquiteto chefe ou central atua como uma ñarquitetura ambulanteò: ele comunica a arquitetura para os desenvolvedores e, por sua vez, comunica os problemas encontrados para os *stakeholders* corretos durante o desenvolvimento. Resumindo: as práticas de arquitetura de software encontradas por elas enfatizam comunicação ao invés de

documentação. Entretanto, é importante ressaltar que o estudo delas focou no trabalho dos arquitetos de software apenas. Com isso, o estudo realizado neste trabalho estende a pesquisa de Unphon e Dittrich (2010), uma vez que em todas as empresas pesquisadas observou-se essa mesma preferência por comunicação em vez de documentação não apenas para os arquitetos de software, mas para arquitetos de TI em geral. Neste caso, com a divisão em tipos de arquiteto, os vários arquitetos também dividem esse papel de ñarquitetura ambulanteò, uma vez que cada um deles foca em um aspecto (ou esfera de conhecimento) da arquitetura de TI. Assim, o arquiteto *enterprise* desempenha o papel de ñarquitetura ambulanteò do ponto de vista organizacional, o de soluções do ponto de vista funcional e o arquiteto de software do ponto de vista técnico.

Este aspecto de os arquitetos serem a principal fonte de informações no desenvolvimento de sistemas também está relacionado com o compartilhamento de conhecimento através das fronteiras. Carlile (2003) afirma que as três primeiras características da Figura 5.2 são similares a características de *boundary objects*. As características são: (i) prover aos atores um conhecimento comum através da transferência de conhecimento nas fronteiras, (ii) estabelecer uma semântica comum para identificar e traduzir diferenças e dependências e (iii) obter o conhecimento de diferentes atores, estabelecer um conhecimento comum para todos no projeto e usá-lo para negociar e transformar o conhecimento dos atores, resolvendo interesses diferentes que bloqueiam o compartilhamento e avaliação do conhecimento. Com isso, ele ressalta a importância dos *boundary objects* nas fronteiras pragmáticas, porém ele menciona como *boundary objects* apenas documentos, protótipos e metodologias. Grinter (1999), por outro lado, argumenta que os arquitetos de software são eles próprios *boundary objects*, pois eles interagem com grupos diferentes de *stakeholders* e passam informações para todos eles, contribuindo assim para o conhecimento comum do projeto. Assim, isso também se alinha com o trabalho de Unphon e Dittrich (2010) e o ñconceitoò de que o arquiteto atua como uma ñarquitetura ambulanteò.

Considerando que arquitetos são *boundary objects*, a divisão de tipos de arquitetos citada neste trabalho cria vários *boundary objects* no projeto (uma vez que cada arquiteto é um). Estes, juntos, organizam o fluxo de informações e compartilhamento de conhecimento dentro do projeto, uma vez que atuam em todas as etapas para transformar as necessidades dos clientes em termos técnicos que os desenvolvedores possam usar. Em outras palavras, os arquitetos (de todos os tipos) se tornam os principais *boundary objects* no processo de desenvolvimento de sistemas de software, uma vez que eles são os *boundary objects* mais

ñacessadosò pelos *stakeholders*, já que estes preferem perguntar a eles a olhar em um documento.

5.3.6. Ferramentas

Existem vários mecanismos voltados para apoiar as atividades dos arquitetos de TI de um modo geral. A maioria das ferramentas foca em um único tipo de arquiteto ou esfera de conhecimento. Quando as ferramentas oferecem apoio a mais de um tipo de arquiteto, isto é feito de forma isolada, sem considerar as interconexões entre eles. Entretanto, os resultados desta pesquisa sugerem que estes mecanismos também precisam focar nas interconexões e dependências entre os diferentes tipos de arquiteto para assim apoiar apropriadamente o fluxo de informações e a transferência, tradução e transformação do conhecimento através das diferentes fronteiras nas quais os arquitetos trabalham. Esse foco nas interconexões e possíveis formas de apoiá-las configura uma área promissora de pesquisa que pode desenvolver ou adaptar mecanismos (metodologias, ferramentas, *frameworks* etc) que auxiliem na colaboração entre os tipos de arquitetos e, consequentemente, contribuam para um melhor compartilhamento de conhecimento entre eles e entre todos os *stakeholders*, de acordo com o que foi discutido na seção 5.3.1.

Por outro lado, uma área de pesquisa recente de arquitetura de software foca no compartilhamento de conhecimento sobre arquitetura de software principalmente sob o ponto de vista de documentação (Ali Babar *et al.*, 2007). As razões para essa abordagem residem no fato de que as arquiteturas de software (ou de TI como discutido aqui) englobam a *expertise* de muitos *stakeholders* diferentes, uma vez que elas são criadas colaborativamente e, por isso, especialmente quando documentadas, podem ser usadas em projetos posteriores como fonte de conhecimento. Além disso, alguns autores (Farenhorst e van Vliet, 2009)(Ali Babar *et al.*, 2007) afirmam que uma estratégia híbrida ï combinando tanto ferramentas formais quanto informais ï é a abordagem mais efetiva para compartilhamento de conhecimento sobre a arquitetura de software. Entretanto, mais uma vez, o foco desta área de pesquisa é a arquitetura de software como algo independente, como se não fosse influenciada pela arquitetura de TI na qual está contida; o que não é o caso como indicam os resultados deste estudo. Outro ponto importante é que, como visto na seção anterior, na indústria não é dada tanta ênfase em documentação, dessa forma as ferramentas adquirem uma importância ainda maior, pois precisam auxiliar nesse processo de compartilhamento de conhecimento. Os

resultados desta pesquisa sugerem que ferramentas informais (como wikis) são de fato muito úteis para arquitetos de TI (e, consequentemente, de software).

Durante a pesquisa identificou-se que algumas empresas usam algumas ferramentas para apoiar o trabalho dos arquitetos, mas, além da maioria delas não ser voltada para arquitetura em si, geralmente elas não atendem a todas as necessidades das empresas ou a própria empresa precisa customizar uma ferramenta e usá-la juntamente com outras ferramentas. Por exemplo, a Empresa A usa a Ferramenta A, que é uma ferramenta de gerência de testes que oferece funcionalidades de garantia de qualidade, incluindo gerência de requisitos e de testes, entre outros. A Empresa A a utiliza para atividades relacionadas com arquitetura, como as revisões de requisitos mencionadas anteriormente e, apesar de esta ferramenta oferecer apoio para esta atividade, ainda assim os informantes argumentam que esta ferramenta ainda não é suficiente (ver seção 4.5).

Já a Empresa D usa uma ferramenta de gerência de projetos e *bugtracking*. Entretanto, a empresa precisou adaptar o sistema de *tracking* da ferramenta para apoiar as atividades de arquitetura, especialmente aquelas relacionadas aos requisitos. Estes *trackers* customizados são utilizados pelos dois tipos de arquiteto que esta empresa apresenta: arquiteto de soluções e arquiteto de software. Por exemplo, se um novo requisito de arquitetura surge, o arquiteto de soluções realiza suas atividades (relacionadas com uma definição geral de como este novo requisito afeta a visão geral da arquitetura) e usa o *tracker* para delegar a tarefa (relacionada ao refinamento do requisito) para o arquiteto de software, que por sua vez vai realizar suas atividades e então passar o desenvolvimento para os desenvolvedores.

A Empresa E utiliza a Ferramenta E, que é uma ferramenta de diagramação, a qual, novamente, não é uma ferramenta específica para arquitetura. Esta empresa usa diferentes ferramentas, normalmente simples, aliadas para suprir as suas necessidades: a Ferramenta E (para a construção de fluxogramas), Powerpoint, quadro branco e wiki, por exemplo.

A Empresa C, por sua vez, está adotando uma ferramenta voltada para arquitetura. De acordo com a especificação desta ferramenta, ela tem funcionalidades relacionadas à gerência de requisitos, modelagem de processo de negócio e apoio ao uso de *frameworks* de arquitetura *enterprise* (entre outros). Porém, no guia de seleção de ferramentas de arquitetura *enterprise* de Schekkerman (2009) esta ferramenta é listada como focando apenas em atividades de arquitetura de soluções e com algum apoio à atividades de arquitetura *enterprise* e de software, não compreendendo outros aspectos usados na classificação deste guia (observância à governança e riscos, gerência de programa, gerência de portfólio *enterprise*/de TI e

estratégia de negócio/TI). Ou seja, apesar desta ser uma ferramenta voltada para arquitetura, seu foco está principalmente no que a academia normalmente define como arquitetura de software e, apesar de até possuir funcionalidades mais voltadas para a visão organizacional (modelagem de processo de negócio e apoio ao uso de *frameworks* de arquitetura *enterprise*), estas continuam sem focar nas interconexões do trabalho dos tipos diferentes de arquitetos.

Outro aspecto identificado durante a pesquisa é o uso de wikis. Em geral, observou-se que diversas empresas utilizam wikis e que elas oferecem um mecanismo fácil para prover acesso e modificações nas informações sobre arquitetura (incluindo seus requisitos). Entretanto, sempre existe o risco de as pessoas não a usarem, ou seja, as pessoas podem não acessar ou modificar as informações salvas na wiki (ver seção 4.4.2). A Empresa D usa um *tracker* customizado que pode em parte preencher essa lacuna, mas ainda assim é dependente de os *stakeholders* o usarem adequadamente. De qualquer forma, com estes *trackers* pode-se diminuir o problema das pessoas não saberem quando e onde as informações foram postadas na wiki.

Apesar da grande utilização de wikis em muitas das empresas pesquisadas e delas serem bem aceitas pelos próprios arquitetos, é importante ressaltar que elas também não focam nas interconexões entre os tipos de arquiteto. Aparentemente uma das vantagens das wikis, e talvez um dos motivos pelo qual muitas empresas as adotam, é a sua flexibilidade, ou seja, as wikis não são associadas a nenhuma linguagem de modelagem específica nem com qualquer formato particular de documentos. Portanto, elas são flexíveis o suficiente para permitir que qualquer tipo de arquiteto passe qualquer informação que ele acredita ser relevante para qualquer outro arquiteto ou *stakeholder*. Outra vantagem está no fato de wikis serem plataformas leves e versáteis (como sugeridas por Farenhorst e Van Vliet (2009) para arquitetos de software e aqui estendidas para arquitetos de TI como um todo), as quais se adaptam melhor à atividades técnicas e não técnicas.

Outro aspecto identificado diz respeito à interação entre negociação de requisitos e *design* da arquitetura. Essas tarefas estão fortemente relacionadas, mas normalmente existe uma separação (conceitual e temporal) destas atividades, o que significa que a pesquisa nas duas áreas também ocorre de forma separada (Kazman e Chen, 2005). Esta separação pode explicar porque os informantes reclamaram de revisões de requisitos ou porque as empresas customizam ferramentas de *tracking*. Isso também ressalta a necessidade de uma ferramenta que ofereça um mecanismo para conectar as atividades de gerência de requisitos e com as atividades de arquitetura de TI. Normalmente, os arquitetos de soluções(ou quem exerce as

atividades deles) são as pessoas que elicitam os requisitos (junto com analistas), mas eles não se preocupam com detalhes técnicos, pois este não é o trabalho deles. Eventualmente, quando os arquitetos de software (que já transformaram essa informação de entrada dos arquitetos de soluções) repassam esses requisitos para os desenvolvedores, eles precisam considerar os aspectos técnicos. Em resumo, os requisitos passam por diversas modificações até chegarem aos desenvolvedores. Assim, se estes requisitos não forem adequadamente gerenciados, vários problemas especialmente relacionados a retrabalho podem ocorrer. Por isso, as atividades de gerência de requisitos devem estar conectadas com as atividades de arquitetura de todos os tipos de arquitetos e essa relação deve ser apoiada por ferramentas apropriadas.

Com isso, tomando como base as empresas pesquisadas e a forma como elas realizam suas atividades de arquitetura, os resultados sugerem que uma estratégia híbrida é sim uma forma efetiva de compartilhamento de conhecimento. Além disso, é possível sugerir um misto de mecanismos composto por wikis, *trackers* para atividades de arquitetura e requisitos, uma ferramenta que apoie a gerência dos requisitos durante todo o processo de desenvolvimento (com especial atenção para as atividades de arquitetura) e reuniões multidisciplinares frequentes (sempre envolvendo arquitetos) como uma abordagem que oferece um bom suporte ao compartilhamento de conhecimento através das fronteiras e, consequentemente, às atividades dos vários tipos de arquitetos. Explicando melhor: as wikis são ferramentas leves e flexíveis, onde qualquer informação sobre o projeto pode ser armazenada sem que esta seja dependente de nenhuma tecnologia/formato/metodologia. Os *trackers* ajudam a resolver o problema de os *stakeholders* não usarem a wiki corretamente e também podem influenciar nas revisões, pois podem ser usados para contatar revisores já fornecendo várias informações necessárias a eles para as revisões. Uma ferramenta de gerência de requisitos que permeie as atividades durante o projeto todo seria ideal para aliar gerência de requisitos e arquitetura e assim ajudar a preencher a lacuna apontada por Kazman e Chen (2005). Por fim, as reuniões multidisciplinares são um meio excelente para apoiar a transferência, tradução e transformação do conhecimento, pois ocasionam discussões com *stakeholders* de vários domínios que auxiliam os arquitetos a transformar o conhecimento entre as principais fronteiras (gerencial, funcional e técnica).

CAPÍTULO 6

6. Conclusões e Trabalhos Futuros

Este trabalho descreveu um estudo qualitativo sobre as atividades dos arquitetos de TI na indústria. Nele identificou-se que as empresas tendem a possuir diferentes tipos de arquitetos, mas que estes são organizados de forma similar. O Capítulo 4 descreveu diversos aspectos relacionados a essa divisão, como a classificação das empresas, requisitos e a tradução de informações. Após isso, no Capítulo 5, a divisão foi analisada e comparada a um *framework* de gerência de conhecimento, através do qual se identificou que os arquitetos desempenham um papel muito importante no compartilhamento de conhecimento dentro dos projetos e que a própria divisão de atividades entre eles favorece este processo. Este último capítulo traz primeiramente as considerações finais sobre este assunto, descrevendo depois as contribuições desta pesquisa e os trabalhos futuros derivados dela.

6.1. Arquitetos e compartilhamento de conhecimento

Conforme apresentado nos capítulos anteriores, a divisão em tipos de arquitetos representa uma forma de dividir tanto as atividades de arquitetura quanto a comunicação, uma vez que cada arquiteto interage com grupos de *stakeholders* específicos. Além disso, também é importante notar que os arquitetos interagem entre si: as atividades deles possuem interconexões e eles dependem do trabalho uns dos outros de formas diferentes. Mais especificamente, eles recebem informações de grupos diferentes de *stakeholders*, realizam suas atividades usando estas e outras informações próprias e então passam outro tipo de informação adiante, inclusive para o próximo arquiteto. Assim, os arquitetos trabalham tanto como agregadores quanto como difusores de informações entre os diversos grupos de *stakeholders*.

Isto se torna ainda mais importante ao considerar as diferentes fronteiras de conhecimento descritas por Carlile (2003). Como explicado no capítulo de discussão, as fronteiras surgem devido à existência de conhecimentos especializados em domínios diferentes. Considerando os principais grupos de *stakeholders* que interagem durante o desenvolvimento de software é possível assumir que cada grupo representa um domínio diferente com um conhecimento especializado diferente, por exemplo, os clientes possuem o conhecimento do seu negócio, os desenvolvedores o conhecimento técnico e o time de

marketing o conhecimento do mercado. Como cada um desses conhecimentos especializados configura um domínio e cada um desses domínios gera uma fronteira de conhecimento, durante o desenvolvimento de sistemas várias fronteiras precisam ser superadas e os arquitetos são um importante meio de se conseguir isso.

Analisando os principais domínios de conhecimento, percebeu-se que eles podiam ser agregados em 3 esferas de conhecimento principais: (i) organizacional, que contém informações gerenciais, financeiras, sobre a organização e de *marketing*; (ii) funcional, que contém as informações dos clientes e analistas; e (iii) técnico, que é formada pelo conhecimento técnico relacionado aos desenvolvedores. Cada esfera de conhecimento representa um grande domínio de conhecimento que contém outros domínios menores, por exemplo, a esfera organizacional contém os domínios do cliente, da gerência e do *marketing*. Comparando a divisão de tipos de arquitetos com estes principais domínios de conhecimento percebe-se claramente que cada arquiteto trabalha em um domínio específico: o arquiteto *enterprise* trabalha no domínio organizacional, o arquiteto de soluções trabalha no domínio funcional e o arquiteto de software trabalha no domínio técnico.

Considerando estes três grandes domínios de conhecimento, os tipos de arquitetos e suas interações, nota-se que os arquitetos são fundamentais para o compartilhamento de conhecimento dentro dos projetos qualquer que seja o tipo de fronteira que exista entre os domínios: sintática, semântica ou pragmática (Carlile, 2003). Na fronteira sintática os arquitetos contribuem com a transferência de conhecimento utilizando uma sintaxe padrão para o projeto, por exemplo, definindo termos comuns que serão usados em todo o projeto. Quando a fronteira é semântica, a contribuição dos arquitetos consiste em realizar a tradução do conhecimento entre domínios diferentes. Isto está relacionado com a seção 4.6 do Capítulo 4 sobre a tradução das necessidades dos clientes para os termos técnicos que os desenvolvedores podem utilizar. Por fim, quando a fronteira é pragmática a divisão de papéis em si se torna mais importante, pois os domínios apresentam conhecimentos mais especializados e construir o conhecimento comum sobre o projeto passa também a ser uma questão política, como por exemplo quando é necessário negociar sobre aspectos como desempenho e segurança caso estes sejam conflitantes em um projeto específico (para obter um maior desempenho é necessário desativar um mecanismo de segurança). Dessa forma, a presença de um tipo de arquiteto em cada um dos principais domínios e, especialmente, a interação entre esses arquitetos contribui bastante para o compartilhamento de conhecimento, uma vez que os arquitetos juntos realizam a transformação do mesmo.

Neste trabalho, foi possível também comparar esses domínios de conhecimento com a classificação das empresas quanto à presença dos diferentes tipos de arquiteto, identificando possíveis motivos pelos quais as diferentes empresas adotam cada configuração (seção 5.3.2, Capítulo 5). Por exemplo, os dados apontam que empresas maiores, com desenvolvimento de vários projetos em paralelo e uso de DDS se beneficiam da divisão em três tipos de arquitetos, pois cada um deles irá focar em um domínio de conhecimento diferente e através de suas interações os conhecimentos dos principais domínios serão integrados para produzir o sistema.

Por fim, aspectos de documentação, ferramentas e requisitos foram analisados considerando os aspectos de compartilhamento de conhecimento mencionados acima. Sobre a documentação de arquiteturas, notou-se que há um esforço maior em comunicação do conhecimento do que em documentação da mesma, sendo que os arquitetos se tornam grandes responsáveis por esta comunicação referente à arquitetura. Sobre as ferramentas identificou-se que, apesar de existirem várias ferramentas/metodologias/*frameworks* que visam apoiar o trabalho dos arquitetos, geralmente estas focam em apenas um tipo de arquiteto ou, quando focam em mais de um, não consideram as importantes interdependências que existem no trabalho deles. Com isso, identificou-se que algumas empresas vêm utilizando uma abordagem híbrida de mecanismos formais e informais para apoiar o trabalho dos arquitetos. Com base nesse fato sugeriu-se que as abordagens utilizadas compreendam wikis, mecanismos de *trackers*, gerência de requisitos e reuniões multidisciplinares. Finalmente, os principais problemas relacionados com o aspecto de requisitos foram identificados e discutidos, concluindo que a presença de arquitetos durante todas as etapas de desenvolvimento de sistema também pode ser útil para evitar que requisitos sejam mal compreendidos e por isso desenvolvidos erroneamente.

6.2. Contribuições do trabalho

Como citado anteriormente, o currículo da SBC para os cursos de Ciência da Computação e Sistemas de Informação apresenta apenas aspectos relativos à arquitetura de software, e nem chega a definir um tópico específico para o assunto. Dessa forma, esse currículo não enfatiza o contexto no qual a arquitetura de software está inserida, a arquitetura de TI de uma organização, o que pode gerar a falsa impressão de que a arquitetura de software não recebe influência de aspectos não técnicos (como os organizacionais). Entretanto, arquitetura de software não é uma atividade isolada, ela precisa ser estudada, entendida, apoiada por

ferramentas que considerem o contexto mais amplo no qual ela está inserida, a arquitetura de TI. Além disso, o trabalho dos arquitetos de software é influenciado por diversos aspectos e diferentes *stakeholders*. Considerando isso, a questão de pesquisa que buscou-se responder com este trabalho era a seguinte: como os arquitetos de TI realizam suas atividades de arquitetura na indústria? Tal questão foi respondida ao longo de todo o trabalho, especialmente nos Capítulos 4 e 5. Neles foi possível identificar que os arquitetos tendem a realizar conjuntos de atividades diferentes que podem ou não ser definidos pela empresa (seção 4.2 e subseções) e que eles representam um importante papel para o compartilhamento de conhecimento no projeto (seção 5.3.1). Identificou-se também quais são as principais interações de cada tipo de arquiteto e as interconexões entre eles próprios (seções 4.3 e subseções). Por fim, as implicações destas observações também foram identificadas e discutidas no contexto de ferramentas, documentação e requisitos relacionados ao trabalho dos arquitetos (seções 5.3.2, 5.3.3, 5.3.4).

Este trabalho também apresenta contribuições ao estender trabalhos anteriores sobre a forma como os arquitetos facilitam a coordenação de atividades (Grinter, 1999)(Unphon e Dittrich, 2010)(Ovaska *et al.*, 2003)(Bass *et al.*, 2007). Trabalhos anteriores também são estendidos no que diz respeito à participação dos arquitetos em todos os estágios do processo de desenvolvimento de software (Taylor *et al.*, 2010)(Hofstader, 2008). Neste trabalho encontrou-se que a divisão em tipos de arquitetos favorece esta prática, pois se houver um arquiteto em cada uma das principais esferas de conhecimento (organizacional, funcional e técnica), sempre haverá ao menos um arquiteto em cada etapa do desenvolvimento de software.

Uma das contribuições pretendidas é a de identificar como o trabalho dos arquitetos de TI na indústria pode influenciar a pesquisa e o ensino em arquitetura de software. Tal contribuição também foi alcançada, pois vários aspectos do trabalho destes profissionais que podem impactar na pesquisa em arquitetura de software foram identificados. Por exemplo, a existência de diferentes tipos de arquitetos que dividem as atividades de arquitetura, a relação entre estes tipos de arquitetos e o compartilhamento de conhecimento, as interconexões entre os arquitetos, os mecanismos de apoio às atividades de arquitetura, entre outros. Esses aspectos, se levados de volta para a academia, podem auxiliar na formação de profissionais de arquitetura que estejam mais próximos da realidade do mercado. Com isso, este trabalho também contribui para a própria indústria, pois se os profissionais forem melhor treinados na

academia, ao passar para o mercado o período de adaptação será menor e eles serão capazes de suprir as demandas da indústria.

6.3. Limitações

Como em todo estudo, a pesquisa aqui descrita apresenta algumas limitações. Primeiramente, há a limitação do método escolhido para coleta de dados, as entrevistas. Em uma entrevista, o informante relata as suas impressões, pensamentos e conclusões, porém é possível que esse relato não reflita as atividades que ele realmente realiza. Isso pode acontecer tanto pelo fato de o informante acreditar que algum ponto não é relevante ou por se tratar de um conhecimento tácito, ou mesmo pelo fato de o informante ter algum receio de que esteja fazendo algo errado e assim relata as atividades que se espera que ele realize.

Além disso, ainda há a limitação característica de estudos qualitativos com relação à generalização. A pesquisa foi conduzida em nove empresas, sendo que em duas não foram obtidas informações conclusivas. Portanto, tem-se um universo de pesquisa de sete empresas, todas localizadas no mesmo país, logo os resultados encontrados são característicos desse universo de pesquisa e seria muito difícil generalizá-los. Isso está de acordo com os objetivos de um estudo qualitativo que consistem de prover um conhecimento profundo sobre uma população em particular, gerando assim hipóteses. Com isso, o que se obtém, além das conclusões para este universo de estudo, são indícios que podem ser investigados em universos maiores de pesquisa para avaliar se o mesmo ocorre em outras realidades. Entretanto, é importante reafirmar que as empresas pesquisadas são bastante diferentes, mesmo estando inseridas na realidade do mesmo país, e isso reforça a importância dos resultados encontrados.

6.4. Trabalhos Futuros

Como trabalho futuro é interessante realizar entrevistas em outras empresas desenvolvedoras de sistemas (e inclusive tentar voltar às Empresas B e H, das quais não se obteve informações suficientes) para adquirir perspectivas diferentes das atividades de arquitetura e dos tipos de arquitetos que possam corroborar ou mesmo questionar os dados apresentados e discutidos aqui. Ainda nesse sentido, é interessante voltar às empresas já pesquisadas para entrevistar atores que realizem as atividades de tipos de arquitetos diferentes daqueles já entrevistados. Por exemplo, na Empresa C só foram entrevistados arquitetos similares aos arquitetos de

soluções, dessa forma, seria interessante em trabalhos futuros entrevistar projetistas, que são os atores similares aos arquitetos de software nesta empresa.

Além disso, uma das principais contribuições deste trabalho para a comunidade científica foi a identificação de oportunidades de pesquisa. Os aspectos abordados na seção 5.3.3 do capítulo anterior (DDS, a presença do arquiteto de negócios, a diferença entre projetos de produtos e de serviços e o uso de métodos ágeis) correspondem interessantes áreas de pesquisa e aspectos que podem contribuir para este trabalho. Por exemplo, identificou-se que, além de poucos estudos focarem no trabalho dos arquitetos na prática, menos ainda consideram a divisão em tipos de arquitetos e, principalmente, as interdependências e interconexões entre eles. Dessa forma, focar nas interdependências e interconexões entre os diferentes tipos de arquitetos configura um trabalho futuro relacionado a esta pesquisa. Por exemplo, pode ser realizado um estudo sobre as forma como eles trocam informações, de forma a identificar oportunidades de melhoria e sugerir abordagens para elas: com estas informações poderia ser possível sugerir uma ferramenta, ou mesmo agregar um conjunto de ferramentas, que integrasse os tipos de informação necessários para vários tipos de arquitetos e favorecesse a interação entre eles.

Outro aspecto interessante de ser explorado futuramente é o que ocorre com as interações entre os arquitetos quando eles estão distribuídos em diferentes localidades, como na Empresa A, visando identificar se os desafios do trabalho dos arquitetos são intensificados e como eles lidam com isso. Observações iniciais sugerem que a própria divisão em tipos de arquitetos contribui para uma melhor coordenação entre as diferentes localidades quando há pelo menos um arquiteto em cada uma delas. Além disso, a localização de cada tipo de arquiteto parece ser importante para a coordenação das atividades deles. Por exemplo, o arquiteto de soluções estar localizado na empresa do cliente pode ser uma boa estratégia para informar às outras localidades sobre as necessidades do cliente que serão atendidas pelo sistema produzido, uma vez que ele irá projetar a arquitetura do sistema como um todo. Como não foi possível obter dados conclusivos sobre este aspecto, seria muito interessante abordar este assunto em trabalhos futuros.

REFERÊNCIAS

AGILE MANIFESTO. Manifesto for Agile Software Development. 2001. Disponível em <http://agilemanifesto.org/:> Acesso em: 27 de maio de 2011.

AKENINE, D.**A Study of Architect Roles by IASA Sweden**. The Architecture Journal ï The Role of an Architect, journal 15, pp. 22-25, 2008.

ALI BABAR, M., DE BOER, R., DINGSØYR, T., FARENHORST, R. **Architectural Knowledge Management Strategies: Approaches in Research and Industry**. Workshop on Sharing and Reusing architectural Knowledge, SHARKô07, 2007.

BANKS, S., LOUIE, E., EINERSON, M. **Constructing Personal Identities in Holiday Letters**. Journal of Social and Personal Relationships, Vol. 17(3): 299ï 327, 2000.

BASS, L., CLEMENTS, P., KAZMAN, R. **Software Architecture In Practice**: Addison-Wesley Professional, 2003.

BASS, M., MIKULOVIC, V., BASS, L., HERBSLEB, J., CATALDO, M. **Architectural Misalignment: An Experience Report**. Proceedings of the Working IEEE/IFIP Conference on Software Architecture, 2007.

BERENBACH, B. **The Other Skills of the Software Architect**. Proceedings of the first international workshop on Leadership and management in software architecture (ICSE), 2008.

BROOKS JR., F. P. **The Mythical Man-Month: Essays on Software Engineering**, 20th Anniversary Edition. Reading, MA: Addison-Wesley, 1995.

CARLILE, P. **Transferring Translating and Transforming: An Integrative and Relational Approach to Sharing and Assessing Knowledge across Boundaries**, Organization Science, 2003.

CARLILE, P., REBENTISCH, E. **Into the Black Box: The Knowledge Transformation Cycle, Management Science**, vol. 49, No. 9, PP. 1180-1195, 2003.

CARRARO, G., CHONG, F. **Software como Serviço (SaaS): uma perspectiva corporativa**, Disponível em: http://msdn.microsoft.com/pt-br/library/aa905332.aspx. Acesso em 01 de dezembro de 2011.

CLEMENTS, P., KAZMAN, R., KLEIN, R. **Working session: Software Architecture Competence**. Proceedings of the Working IEEE/IFIP Conference on Software Architecture (WICSA'07), 2007.

CHRISTENSEN, H. B., HANSEN, K. M., SCHOUGAARD, K. R. **An Empirical Study on Software Architectsô Concerns**.16th Asia-Pacific Software Engineering Conference, 2009.

CNNMONEY.COM. **Best Jobs in America ï Top 100**. 2010. Disponível em: <http://money.cnn.com/magazines/moneymag/bestjobs/2010/full_list/index.html>. Acesso em 23 de novembro de 2010.

CONWAY, M. E. **How Do Committees invent?** Datamation, 14(4), 28-31, 1968.

CORBUCCI, H., GOLDMAN, A., KATAYAMA, E., KON, F., MELO, C. O. **Genesis and Evolution of the Agile Movement in Brazil ï Perspective from Academia and Industry**. Simpósio Brasileiro de Engenharia de Software (SBES 2011), São Paulo, 2011.

CURTIS, B., KRASNER, H., ISCOE, N. **A Field Study of the Software Design Process for Large Systems**. Communications of the ACM, vol. 31, n. 11, 1988.

DE SOUZA, C. R. B. ; REDMILES, David F. **On the Alignment of Organizational and Software Structure**. In: Brian Whitworth; Aldo de Moor. (Org.).Handbook of Research on Socio-Technical Design and Social Networking Systems. 1 ed. : IGI Global publication, 2008, v. 1, p. 1-13.

DEWALT, K., DEWALT, B. **Participant Observation - A Guide for Fieldworkers**. AltaMiraPress, Walnut Creek, CA, 2002.

DYBÅ, T., DINGSØYR, T. **Empirical studies of agile software development: A systematic review**. Information and Software Technology, 50(9-10): 833-859, 2008.

FARENHORST, R., VAN VLIET, H. **Understanding How to Support Architects in Sharing Knowledge**. Workshop on Sharing and Reusing architectural Knowledge, SHARK'09, 2009.

FOWLER, M. **Patterns of Enterprise Application Architecture**: Addison-Wesley Longman Publishing Co., Inc, 2002.

GRINTER, R. **Systems architecture: product designing and social engineering**. Proceedings of the International Conference on Work activities Coordination and Collaboration, 1999.

GOLDMAN, A., KATAYAMA, E. **Retrato da comunidade acadêmica de métodos ágeis no Brasil**. 2010. Disponível em: <http://www.agilcoop.org.br/files/pesquisadores.de_.metodos.ageis_.pdf>. Acesso em: 01 de novembro de 2011.

HERBSLEB J., GRINTER, R. **Architectures, Coordination, and Distance: Conway's Law and Beyond**. IEEE Software, Setembro/Outubro, PP. 63-70, 1999.

HERBSLEB, J., MOCKUS, A., FINHOLT, T., GRINTER, R. **An empirical study of global software development: Distance and speed**. In Proceedings of the International Conference on Software Engineering, Toronto, Canada, May 15-18, pp. 81-90, 2001.

HOFSTADER, J. **We donôt need no architects!** The Architecture Journal ï The Role of an Architect, journal 15, pp. 2-6, 2008.

IASA.**IASA ï An Association for all IT architects**. Disponível em: <http://www.iasaglobal.org/iasa/Default.asp>. Acesso em: 04 de abril de 2011.

KAZMAN R., IN, H., CHEN, H. **From requirements negotiation to software architecture decisions**, In Information and Software Technology 47 (2005) 511ï 520, 2005.

KRALJ, M. **The need for an Architectural Body of Knowledge**. The Architecture Journal ï The Role of an Architect, journal 15, pp. 17-20, 2008.

MORNEAU, K. A., TALLEY, S. **Architecture: an emerging core competence for IT professionals**, 2011. Anais da 8ª ACM SIGITE Conference on Information Technology Education, 2007.

OVASKA, P., ROSSI, M., MARTTIIN, P. **Architecture as a coordination tool in multi-site software development**. Software Process: Improvement and Practice 8(4): 233-247, 2003.

PARNAS, D. L. **On the Criteria to be used in decomposing systems into modules**. Communications of the ACM, 15(12), 1053-1058, 1972.

PEREIRA, M. Z. **PSOA - um framework de práticas e padrões SOA para projetos DDS**. Março de 2011. Dissertação (Mestrado) - Faculdade de Informática, PUCRS. Porto Alegre, 2011.

PEREIRA, M. Z. ; AUDY, J. L. N. ; PRIKLADNICKI ; FIGUEIREDO, M. C. ; SOUZA, C. R. B. **SOA practices and patterns applied in global software development**. In: International Conference on Enterprise Information Systems, 2011, Pequim. Proceedings of the International Conference on Enterprise Information Systems, 2011.

PREISS, P. **Architecture Journal Profile: Paul Preiss**. The Architecture Journal ï The Role of an Architect, journal 15, pp. 10-12, 2008.

SCHEKKERMAN, J. **Enterprise Architecture Tool Selection Guide**, 2009. Disponível em: <http://www.enterprise-architecture.info/EA_Tools.htm>. Acesso em 20 de fevereiro de 2011.

SBC. **Currículo de Referência da SBC para Cursos de Graduação em Computação e Informática**. 1999. Disponível em: http://www.sbc.org.br/index.php?option=com_jdownloads&Itemid=195&task=viewcategory&catid=36. Acesso em 02 de maio de 2011.

SEAMAN, C. **Qualitative Methods**. In: Guide to Advanced Empirical Software Engineering. Londres: Springer-Verlag, 35-62, 2008.

SHAW, M., HERBSLEB, J., OZKAYA, I. Deciding what to design: closing a gap in software engineering education. ICSE 2005: 607-608.

SHAW, M., VAN VLIET, H. **Software Architecture Education Session Report**. WICSA 2005: 185-190.

SHIREY, J. **The Softer Side of the Architect**. The Architecture Journal ï The Role of an Architect, journal 15, pp. 26-28, 2008.

SINGER, J., SIM, S., LETHBRIDGE, T. **Software Engineering Data Collection for Field Studies**. In: Guide to Advanced Empirical Software Engineering. Londres: Springer-Verlag, p. 9-34, 2008.

SMOLANDER, K. **Four Metaphors of Architecture in Software Organizations: Finding out the Meaning of Architecture in Practice**. Proceedings of the 2002 International Symposium on Empirical Software Engineering (ISESEê02), 2002.

SPANOUDAKIS G., ZISMAN A. **Software Traceability: A Roadmap**. Advances in Software Engineering and Knowdledge Engineering, (ed) S.K Chang, World Scientific Publishing, 2005.

STRAUSS, A., CORBIN, J. **Pesquisa Qualitativa: Técnicas e Procedimentos para o Desenvolvimento de Teoria Fundamentada**. Artmed, 2008.

TAYLOR, R. N., MEDVIDOVIC, N., DASHOFY, E. M. **Software Architecture: Foundations, Theory, and Practice**. Wiley, 2010.

THE OPEN GROUP. **Business Executive's Guide to IT Architecture Have you thought**. Dez. 2004. Disponível em: <http://www.opengroup.org/bookstore/catalog/w043.htm>. Acesso em 15 de novembro de 2010.

THE OPEN GROUP. **IT Architect Certification Program**. Disponível em: <http://www.opengroup.org/itac>. Acesso em 15 de novembro de 2010.

THE OPEN GROUP. **Information Technology Architect Certification Program ï Conformance Requirements (Multi-level)**. Inglaterra, mai. 2008. Disponível em: <http://www.opengroup.org/itac>. Acesso em 15 de novembro de 2010.

UNDE, A. **Becoming an Architect in a System Integrator**. The Architecture Journal ï The Role of an Architect, journal 15, pp. 7-9, 2008.

UNPHON, H., DITTRICH, Y. **Software architecture awareness in long-term software product evolution**. The Journal of Systems and Software, 83, pp 2211ï 2226, 2010.

VERBI Software. **MAXQDA ï The art of text analysis**. Disponível em: http://www.maxqda.com/. Acesso em: 05 jan 2010.

WILT, J. **IASAôs Five Pillars of IT Architecture**. Microsoft tech.ed 2010. Disponível em: <http://www.msteched.com/2010/Europe/ARC204>. Acesso em 16 de novembro de 2010.

APÊNDICE A

Questionário de caracterização das empresas

A intenção das próximas perguntas é obter algumas informações sobre a empresa para caracterizá-la de forma a tentar generalizar os resultados. É importante reafirmar que as informações serão anônimas e não serão divulgados nomes nem da empresa nem dos informantes.

1) Qual o nicho de negócio da empresa?
2) A empresa trabalha com desenvolvimento distribuído de software?
( ) sim ( ) não
3) Os clientes da empresa costumam ser:
( ) internos ( ) externos ( ) ambos
4) Número (aproximado) de clientes da empresa _____
    a. No país: _____
    b. No exterior: _____
5) Quantos projetos aproximadamente a empresa costuma desenvolver por ano?
6) Qual o tamanho aproximado dos projetos que a empresa costuma desenvolver? (na escala que a empresa utilizar, por exemplo: pontos por função, linhas de código)
7) Qual o tempo médio de duração dos projetos (aproximadamente)?
8) Qual o seu grau de concordância com a seguinte afirmação: ñDe um modo geral, os projetos na minha empresa terminam no tempo definidoò.
    ( ) Concordo fortemente.
    ( ) Concordo.
    ( ) Neutro.
    ( ) Discordo.
    ( ) Discordo fortemente.
9) Qual o seu grau de concordância com a seguinte afirmação: ñDe um modo geral, os projetos na minha empresa terminam no custo definidoò.
    ( ) Concordo fortemente.
    ( ) Concordo.
    ( ) Neutro.
    ( ) Discordo.
    ( ) Discordo fortemente.
10) Quantas pessoas normalmente são envolvidas por projeto (aproximadamente)?
11) Quantos funcionários a empresa possui aproximadamente?

12) A empresa já foi avaliada em algum programa de melhoria de qualidade (CMMI ou MPS.Br)? Se sim, qual nível e programa?

( ) sim ( ) não

Programa/Nível: _____

13) Como você identifica o nível de satisfação dos clientes?

( ) Totalmente Satisfeitos ï todos os clientes se demonstraram satisfeitos com os produtos e/ou serviços da empresa.

( ) Largamente Satisfeitos ï a maioria dos clientes se demonstra satisfeita com os produtos e/ou serviços da empresa.

( ) Parcialmente Satisfeitos ï a minoria dos clientes se demonstra satisfeita com os produtos e/ou serviços da empresa.

( ) Não Satisfeitos ï nenhum cliente se demonstra satisfeita com os produtos e/ou serviços da empresa.

( ) Satisfação Desconhecida ï a empresa não conhece o grau de satisfação de seus clientes.

APÊNDICE B

Guia de entrevistas da primeira etapa de coleta de dados

- Perguntas Grais
    1) Por favor, qual seu nome completo?
    2) Há quanto tempo você trabalha na empresa?
    3) Qual o cargo ocupado na empresa?
    4) Há quanto tempo você ocupa este cargo?
    5) Quais os projetos mais importantes em que você está atualmente envolvido?
    6) Quais são as atividades que você realiza como <cargo>?
    7) Quantas pessoas trabalham com você? Qual trabalho que elas realizam?

- Sobre o projeto de DDS
    8) Algum dos projetos que você citou envolvem pessoas em diferentes localidades, isto é, desenvolvimento distribuído? Qual (se for mais de um pedir que descreva um como exemplo)?
        a. Quantas equipes diferentes trabalham nesse projeto?
        b. Onde as equipes estão localizadas?
        c. Como é a divisão de atividades? Que parte do projeto cabe a sua equipe?
        d. Como a parte relativa a sua equipe se relaciona com a das outras equipes?
    9) Caso o informante não esteja envolvido em nenhum projeto de DDS atualmente: quando foi a última vez que você trabalhou em um projeto distribuído?
        a. como arquiteto?
            - caso ele já tenha trabalhado com DDS em menos de 6 anos prosseguir (arquiteto tipo 1)
            - caso não pular para perguntas relacionadas a empresa (arquiteto tipo 2)
    10) Como foi feita a divisão do trabalho entre as equipes?
        a. Quem fez a divisão?
        b. Quando?
        c. Estas equipes já trabalharam juntas antes?

- Sobre o trabalho do entrevistado
    11) Quais as suas atividades específicas para este projeto?
    12) Foi definida uma arquitetura para ser utilizada neste projeto?
    13) Qual o ñprocessoò utilizado para a definição da arquitetura neste projeto?
        a. Qual a sua participação nesta etapa?
        b. Quantas pessoas participam desta etapa? Quem são as outras pessoas? Onde elas estão localizadas?

c. Como a equipe de definição de arquitetura é formada? Como as pessoas dessa equipe são escolhidas?
    i. Pessoas que não são arquitetos participam desta etapa? Membros das diferentes equipes? Clientes? Experts? Desenvolvedores? Indicações?
    ii. Qual o papel de cada membro da equipe na definição da arquitetura?

14) Quando e como a divisão do trabalho entre as equipes foi feita neste projeto?
    a. Foi você quem designou a divisão dos módulos?
    b. Os módulos foram divididos entre as equipes seguindo algum critério? Qual?
    c. Como foi feita a identificação de qual equipe seria melhor para desenvolver qual módulo?
    d. A divisão considerou a arquitetura? De que forma?
    e. A divisão considerou a distribuição? De que forma?
    f. A divisão considerou a estrutura da organização? De que forma?
15) Após a definição da arquitetura, qual o próximo passo? Por exemplo:
    a. Apresenta a arquitetura para o restante das equipes?
    b. Negocia a arquitetura?
    c. Recebe feedback dos outros envolvidos?
    d. Modifica a arquitetura baseando-se no feedback?
16) Como foi/é feita a definição do cronograma neste projeto?
    a. Levou em consideração a distribuição? De que forma?

- Sobre a arquitetura em si

17) Que fatores influenciam na definição da arquitetura / módulos?
18) Você acha que a Arquitetura de Software é um fator crítico de sucesso em um projeto distribuído? Por quê?
19) Quais as principais dificuldades encontradas pela equipe de projeto em relação a arquitetura do software?
20) Em quais etapas de desenvolvimento essas dificuldades foram identificadas?
21) O fato das equipes serem distribuídas afeta a arquitetura?
    a. Se uma equipe é especialista em um tipo de trabalho, funcionalidades relacionadas a ele são agregadas em um módulo?
22) Você pode descrever estratégias utilizadas para distribuir o desenvolvimento de componentes utilizados nesse projeto?
    a. Componentes complexos são divididos? Ou uma única equipe o desenvolve?
    b. Agregam-se funcionalidades em componentes de acordo com a especialidade das equipes? Ou os componentes são definidos e somente depois se pensa na distribuição?
23) Se respondeu afirmativo, a questão 14.e, Você pode me citar um exemplo de uma estratégia utilizada na arquitetura para lidar com a distribuição do desenvolvimento? Algo que poderia ser desenvolvido de outra maneira, mas que para DDS precisou-se pensar em uma saída diferente?
24) Existe alguma estratégia que foi bem sucedida em outros projetos que é frequentemente reutilizada em projetos distribuídos?

- Sobre a interação entre as equipes

25) Como é a dependência entre o trabalho das diferentes equipes?
26) Como as dependências são gerenciadas no projeto?
    a. As interfaces são definidas?

b. Quando e como é feita a integração entre os módulos?
c. Se na hora de integrar os módulos houver algum problema, como isso é resolvido? Isso já aconteceu alguma vez?

27) Quão frequentemente você precisa interagir com pessoas de outras equipes neste projeto (localizadas em outros lugares)?
28) De que forma se dá essa interação?
29) Quão frequentemente as equipes precisam interagir entre si?
30) Você contribui para a interação entre as equipes? De que forma?
31) Como as informações do desenvolvimento de uma equipe são passadas para outra?
    a. Reuniões? E-mails? Relatórios?
32) Se a estrutura de um componente de uma equipe precisa ser modificada, como isso afeta as outras equipes? Como essa modificação é informada para as outras equipes?

- Perguntas sobre a empresa

33) Quais os mercados-alvo dos softwares desenvolvidos pela empresa?
34) Quais são as plataformas / linguagens de desenvolvimento utilizadas pela empresa?
35) De uma maneira geral, em que etapa do ciclo de desenvolvimento do software a empresa costuma trabalhar a arquitetura de software?
36) Você acha que a Arquitetura de Software é um fator crítico de sucesso em um projeto distribuído? Por quê?
37) Que dificuldades poderiam ser encontradas pelas equipes de projeto em DDS em relação a arquitetura do software?
38) Como essas dificuldades poderiam ser minimizadas?
39) Você acha que o tipo de plataforma/linguagem utilizada ou produto/serviço desenvolvido pode influenciar no sucesso de um projeto distribuído? Quais?

APÊNDICE C

Guia de entrevistas da segunda etapa de coleta de dados

- Perguntas gerais
  1) Por favor, qual seu nome completo?
  2) Há quanto tempo você trabalha na empresa?
  3) Qual o cargo ocupado na empresa?
  4) Há quanto tempo você ocupa este cargo?
  5) Quais os projetos mais importantes em que você está atualmente envolvido?
  6) Quais são as atividades que você realiza como <cargo>?
  7) Quantas pessoas trabalham com você? Qual trabalho que elas realizam?

- Sobre projeto de DDS
  8) Algum dos projetos que você citou envolvem pessoas em diferentes localidades, isto é, desenvolvimento distribuído? Qual (se for mais de um pedir que descreva um como exemplo)?
     a. Quantas equipes diferentes trabalham nesse projeto?
     b. Onde as equipes estão localizadas?
     c. Como é a divisão de atividades?
     d. Que parte do projeto cabe a sua equipe?
     e. Como a parte relativa a sua equipe se relaciona com a das outras equipes?
  9) Caso o informante não esteja envolvido em nenhum projeto de DDS atualmente: quando foi a última vez que você trabalhou em um projeto distribuído?
     a. como arquiteto?
        i. caso ele já tenha trabalhado com DDS em menos de 6 anos prosseguir (arquiteto tipo 1)
        ii. caso não pular para perguntas relacionadas a empresa (arquiteto tipo 2)
  10) Como foi feita a divisão do trabalho entre as equipes?
     a. Quem fez a divisão?
     b. Quando?
     c. Estas equipes já trabalharam juntas antes?

- Sobre o trabalho do entrevistado
  11) Quais as suas atividades específicas para este projeto?
  12) Qual o ñprocessoò utilizado para a definição da arquitetura neste projeto?
     a. Qual a sua participação nesta etapa?

b. Quantas pessoas participam desta etapa? Quem são as outras pessoas? Onde elas estão localizadas?
c. Como a equipe de definição de arquitetura é formada? Como as pessoas dessa equipe são escolhidas?
    i. Pessoas que não são arquitetos participam desta etapa?
    ii. Membros das diferentes equipes? Clientes? Experts? Desenvolvedores? Indicações?
    iii. Qual o papel de cada membro da equipe na definição da arquitetura?

13) Quando e como a divisão do trabalho entre as equipes foi feita neste projeto?
    a. A divisão considerou a arquitetura? De que forma?
    b. A divisão considerou a distribuição? De que forma?
    c. A divisão considerou a estrutura da organização? De que forma?

- Sobre o trabalho de arquiteto de software

14) O que você acha que um arquiteto de software precisa saber / conhecer / agir para ser um bom arquiteto de software?
15) Quantos arquitetos normalmente trabalham em um projeto da empresa?
    a. Eles tem diferentes perfis? Quais as especificidades desses perfis?
    b. Quais os benefícios dessa configuração?
    c. Se sim para a), quais as tarefas de cada tipo de arquiteto?
    d. Se sim para a), qual o perfil que você ocupa? Qual o seu trabalho e como ele se relaciona com os outros perfis?
    e. Se sim para a), os arquitetos estão localizados em que sites?
        i. Todos no mesmo local ou distribuídos entre as equipes?
        ii. Algum motivo especial para essa configuração?
16) Como é a interação entre os arquitetos?
    a. Eles trabalham em conjunto?
    b. Como eles interagem durante a definição da arquitetura?
    c. Como eles interagem com os outros *stakeholders*?
17) Como é a participação do arquiteto na comunicação entre as equipes dos diferentes sites? E na comunicação dentro da(s) equipe(s) do seu site?
18) Você se comunica frequentemente com outros tipos de *stakeholders* que não desenvolvedores (ex: clientes, gerentes, marketing)?
    a. Com que frequência?
    b. Quais os motivos?
    c. Apenas do mesmo site que você ou também dos outros sites?
19) Outros *stakeholders* (desenvolvedores, clientes, gerentes etc) vêm até você para tirar dúvidas sobre a arquitetura?
    a. Com que frequência?
    b. Apenas do mesmo site que você ou também dos outros sites?
    c. Existem outras pessoas que podem também tirar essas dúvidas? Quem são e qual o papel delas no projeto?

- Sobre arquitetura de software

20) De que forma a arquitetura é representada?
    a. Texto? Código fonte? Caixas e setas? Classes, pacotes e diagramas?
    b. Diferentes visões?
21) Qual o nível de detalhe da representação da arquitetura?

22) Como essa representação da arquitetura é usada?
    a. Design?
    b. Comunicação entre desenvolvedores?
    c. Comunicação para mudanças?
    d. Comunicação para desenvolver novas funcionalidades?
    e. Comunicação para *bug fixing*?
    f. *Feedback* para implementação?
    g. Distribuição de trabalho e responsabilidades?
23) Como a representação da arquitetura é atualizada?
    a. Nunca? Controlada regularmente? Continuamente? Relacionada ao plano geral ou de releases? Só quando ocorrem problemas?
24) Que fatores influenciam na definição da arquitetura / módulos?
25) Você acha que a Arquitetura de Software é um fator crítico de sucesso em um projeto distribuído? Por quê?
26) Quais as principais dificuldades encontradas pela equipe de projeto em relação a arquitetura do software?
27) Em quais etapas de desenvolvimento essas dificuldades foram identificadas?
28) O fato das equipes serem distribuídas afeta a arquitetura?

- Sobre a interação entre as equipes
29) Como é a dependência entre o trabalho das diferentes equipes?
30) Como as dependências são gerenciadas no projeto?
    a. As interfaces são definidas?
    b. Quando e como é feita a integração entre os módulos?
    c. Se na hora de integrar os módulos houver algum problema, como isso é resolvido? Isso já aconteceu alguma vez?
31) Quão frequentemente você precisa interagir com pessoas de outras equipes neste projeto (localizadas em outros lugares)?
32) De que forma se dá essa interação?
33) Quão frequentemente as equipes precisam interagir entre si?
34) Você contribui para a interação entre as equipes? De que forma?
35) Como as informações do desenvolvimento de uma equipe são passadas para outra?
    a. Reuniões? E-mails? Relatórios?
36) Se a estrutura de um componente de uma equipe precisa ser modificada, como isso afeta as outras equipes? Como essa modificação é informada para as outras equipes?

APÊNDICE D

Guia de entrevistas da terceira e quarta etapas de coleta de dados

- Perguntas gerais
  1) Qual seu nome completo?
  2) Há quanto tempo você trabalha na empresa?
  3) Qual o cargo ocupado na empresa?
  4) Há quanto tempo você ocupa este cargo?
  5) Quais os projetos mais importantes em que você está atualmente envolvido?
  6) Quais são as atividades que você realiza como <cargo>?

- Sobre o trabalho de arquiteto e arquitetura
  7) Quais as suas atividades específicas para este projeto?
  8) Como foi definida a arquitetura neste projeto?
     a. Qual a sua participação nesta etapa?
     b. Quem define a arquitetura?
        i. Pessoas que não são arquitetos participam desta etapa?
        ii. Membros das diferentes equipes? Clientes? Experts? Desenvolvedores? Indicações?
        iii. Qual o papel de cada membro da equipe na definição da arquitetura?
  9) Que fatores influenciam na definição da arquitetura / módulos?
  10) De que forma a arquitetura é representada?
  11) O que você acha que um arquiteto de software precisa saber / conhecer / agir para ser um bom arquiteto de software?
  12) Quais papéis interagem mais frequentemente com o arquiteto?
  13) Quantos arquitetos normalmente trabalham em um projeto da empresa?
     a. Eles tem diferentes perfis? Quais as especificidades desses perfis?
     b. Quais os benefícios dessa configuração?
     c. Se sim para a), quais as tarefas de cada tipo de arquiteto?
     d. Se sim para a), qual o perfil que você ocupa? Qual o seu trabalho e como ele se relaciona com os outros perfis?
  14) Como é a interação entre os arquitetos ou arquitetos e papéis da questão 12?
     a. Eles trabalham em conjunto?
     b. Como eles interagem durante a definição da arquitetura?
     c. Como eles interagem com os outros *stakeholders*?
  15) Como é a participação do arquiteto na comunicação entre as equipes?

16) Você se comunica frequentemente com outros tipos de *stakeholders* que não desenvolvedores (ex: clientes, gerentes, marketing)?
    a. Com que frequência?
    b. Quais os motivos?
17) Outros *stakeholders* (desenvolvedores, clientes, gerentes etc) vêm até você para tirar dúvidas sobre a arquitetura?
    a. Com que frequência?
    b. Existem outras pessoas que podem também tirar essas dúvidas? Quem são e qual o papel delas no projeto?

- Sobre tradução

18) Como é o ñprocessoò de desenvolvimento do projeto desde o pedido do cliente até os desenvolvedores?
    a. Quais artefatos são produzidos?
        i. Qual conteúdo dos artefatos?
19) Falar da teoria de tradução e perguntar se algo assim ocorre na empresa
    a. Como é a divisão de atividades?
    b. Ajuda na difusão de informações?
    c. Quais as vantagens dessa configuração?
    d. Mais alguma informação a acrescentar?

- Sobre dificuldades/problemas/más práticas

20) Quais as principais dificuldades encontradas pela equipe de projeto em relação a arquitetura do software?
21) Já aconteceu de requisitos serem especificados e implementados de acordo com a especificação mas na hora não serem o que o cliente queria?
    a. Por que isso acontece?
    b. Como vocês resolvem? O que acham que ajudaria?
    c. O que vocês esperam do papel do analista nesse sentido?
    d. Vocês sabem qual o treinamento que os analistas recebem? Engenharia de requisitos?
22) Já aconteceu de vocês desenvolverem uma arquitetura que é boa, atende às necessidades, funciona, é elegante, mas se torna muito difícil ou complicada para a equipe desenvolver?
    a. O que é feito quando isso acontece?
    b. Como evitar isso?
23) Más práticas?